The information in this document has been carefully checked and is believed to be entirely reliable. FORCE COMPUTERS makes no warranty of any kind with regard to the material in this document, and assumes no responsibility for any errors which may appear in this document. FORCE COMPUTERS reserves the right to make changes without notice to this, or any of its products, to improve reliability, performance, or design.

FORCE COMPUTERS assumes no responsibility for the use of any circuitry other than circuitry which is part of a product of FORCE COMPUTERS Inc./GmbH. FORCE COMPUTERS does not convey to the purchaser of the product described herein any license under the patent rights of FORCE COMPUTERS Inc./GmbH nor the rights of others. All product names as mentioned herein are the trademarks or registered trademarks of their respective companies.
## Table of Contents

Using This Manual .......................................................... ix

1 Introduction ........................................................................ 1
  1.1 Specification ................................................................. 2
  1.2 Ordering Information .................................................... 6

2 Installation ......................................................................... 7
  2.1 Safety Note ..................................................................... 7
  2.2 Installation Prerequisites and Requirements ....................... 8
    2.2.1 Requirements ..................................................... 8
    2.2.2 Terminal Connection ........................................... 9
    2.2.3 Functional and Location Overview ............................. 9
  2.3 Automatic Power Up – Voltage Sensor and Watchdog Timer .... 11
  2.4 Switch Settings ............................................................ 11
  2.5 Front Panel .................................................................... 15
  2.6 SYS68K/CPU-60 Parameters and 16-bit Timers – CIO ........... 17
  2.7 Serial I/O Ports – SCC .................................................. 17
  2.8 SCSI ............................................................................. 21
  2.9 Floppy Disk – FDC ....................................................... 23
  2.10 Ethernet – LAN .......................................................... 23
  2.11 VMEbus P2 Connector Pinout ........................................ 24
  2.12 SYS68K/IOBP-1 ........................................................ 28
  2.13 Testing the CPU Board Using VMEPROM ....................... 30

3 Hardware .......................................................................... 33
  3.1 SYS68K/CPU-60 Memory Map ........................................ 36
  3.2 SYS68K/CPU-60 Interrupt Map ........................................ 38
## Contents

3.3 **SYS68K/CPU-60 Parameters and Timers – CIO Z8536** ........................................... 40
   3.3.1 MEM-60 DRAM Capacity and CIO1 Timer 3 ........................................... 41
   3.3.2 Flash V_{pp}, Floppy Disk Control, and CIO1 Timer 2 .................................. 42
   3.3.3 MODE x Rotary Switch Setting ..................................................................... 43
   3.3.4 On-board DRAM Capacity and Automatic A24 Expansion ............................ 44
   3.3.5 Board ID and DIAG Display ......................................................................... 44
   3.3.6 A24-to-A32 Address Translation ................................................................... 45

3.4 **68060 CPU** ......................................................................................................... 45
   3.4.1 Hardware Interface of the 68060 CPU ......................................................... 45
   3.4.2 Instruction Set of the 68060 CPU .................................................................. 46
   3.4.3 Vector Table of the 68060 CPU .................................................................... 46

3.5 **Watchdog Timer** ............................................................................................... 48
   3.5.1 Watchdog Operation ...................................................................................... 48
   3.5.2 Watchdog Register Map ................................................................................ 49

3.6 **RIALTO Bus Bridge** .......................................................................................... 50
   3.6.1 Register Set .................................................................................................... 50
   3.6.2 Bridge Configuration Register ...................................................................... 50

3.7 **FGA-002 Gate Array** ........................................................................................ 51

3.8 **DRAM** .............................................................................................................. 52
   3.8.1 Register Set .................................................................................................... 53
   3.8.2 Memory Configuration Register .................................................................... 54
   3.8.3 Memory Diagnostic Register ......................................................................... 55
   3.8.4 DRAM Performance ...................................................................................... 56
   3.8.5 Reading the DRAM Capacity ....................................................................... 57
   3.8.6 DRAM Organization ...................................................................................... 57
   3.8.7 Cache Coherence and Snooping ................................................................... 58
   3.8.8 DRAM Access from the 68060 CPU ............................................................. 59
   3.8.9 DRAM Access via the VMEbus ..................................................................... 60
   3.8.10 DRAM Access from the Ethernet-Controller ............................................... 61
   3.8.11 DRAM Access from the SCSI-Controller .................................................... 61

3.9 **User SRAM (factory option)** ............................................................................. 61
   3.9.1 Backup Power for the User SRAM ................................................................. 62
3.10 System PROM ................................................................. 62
  3.10.1 Device Types for the System PROM ............................... 63
  3.10.2 Address Map of the System PROM ................................. 63
  3.10.3 Reading and Programming the System PROM ....................... 64

3.11 Boot PROM ............................................................... 64
  3.11.1 Boot PROM Address Map and Factory Options ..................... 66
  3.11.2 Programming the Boot PROM ....................................... 67

3.12 User Flash ............................................................... 68
  3.12.1 Programming the User Flash ....................................... 68

3.13 Local SRAM .............................................................. 69
  3.13.1 Local SRAM Organization ........................................... 69
  3.13.2 Devices Types for the Local SRAM ................................ 70
  3.13.3 Backup Power for the Local SRAM ................................. 71

3.14 Real-Time Clock – RTC 72423 ...................................... 71
  3.14.1 RTC Registers Address Map ....................................... 72
  3.14.2 Reading from or Writing to the RTC 72423 ....................... 73
  3.14.3 Backup Power for the RTC 72423 ................................ 74

3.15 VMEbus Interface ....................................................... 75
  3.15.1 Exception Signals SYSFAIL, SYSRESET, and ACFAIL ............ 76
  3.15.2 Master Interface: Address Modifier (AM) Codes .................. 77
  3.15.3 Master Interface: Data Transfer Size ............................. 78
  3.15.4 Master Interface: Burst to VMEbus ................................ 80
  3.15.5 Slave Interface: Access Address ................................. 81
  3.15.6 Slave Interface: DRAM Data Transfer Size ....................... 81
  3.15.7 Slave Interface: Address Modifier Decoding and A24 Slave Mode 81
  3.15.8 Slave Interface: Locked Cycles .................................. 82

3.16 VMEbus Arbitration .................................................... 83
  3.16.1 Single-Level VMEbus Arbiter ...................................... 83
  3.16.2 VMEbus Requester ................................................... 83
  3.16.3 VMEbus Release Modes ............................................. 84
  3.16.4 VMEbus Grant Driver ............................................... 86

3.17 VMEbus Slot-1 .......................................................... 86
3.17.1 Slot-1 (System Controller) Functions ........................................ 87
3.17.2 Slot-1 Detection ........................................................................ 87
3.17.3 Slot-1 Status Register ............................................................... 88
3.17.4 The SYSCLK Driver .................................................................. 88
3.17.5 VMEbus Timer ................................................................. 89

3.18 Serial I/O – SCC AM 85C30 ...................................................... 89
   3.18.1 RS-485 Configuration .............................................................. 90

3.19 SCSI – 53C720SE ................................................................. 92
   3.19.1 SCSI Register Map ............................................................... 93
   3.19.2 Communication across the SCSI bus ....................................... 93

3.20 Floppy Disk – FDC 37C65C ...................................................... 94

3.21 Ethernet – LAN AM 79C965A ................................................... 96
   3.21.1 Register Access ................................................................. 98

3.22 Reset Generation ........................................................................ 101
3.23 Information on Front Panel Devices ........................................... 103

4 Circuit Schematics ........................................................................ paginated separately

5 Data Sheets .................................................................................... paginated separately
   5.1 CIO Z8536
   5.2 FDC 37C65C
   5.3 LAN AM 79C965A
   5.4 RTC 72421
   5.5 SCC AM 85C30
   5.6 SCSI 53C720SE

6 VMEPROM .................................................................................. 125
   6.1 Power-up Sequence ................................................................. 126
   6.2 Front Panel Related VMEPROM Features ................................... 127
      6.2.1 Reset Key ........................................................................ 127
      6.2.2 Abort Key ....................................................................... 127
      6.2.3 Rotary Switches .............................................................. 128
   6.3 Memory Usage of VMEPROM .................................................. 132
Contents

6.3.1 Default Memory Usage of VMEMROM ........................................ 132
6.3.2 Default ROM Use of VMEMROM ........................................... 132

6.4 Devices and Interrupts Used by VMEMROM ........................................ 133
6.4.1 Addresses of the On-Board I/O Devices ..................................... 133
6.4.2 On-Board Interrupt Sources .................................................. 134
6.4.3 Off-Board Interrupt Sources .................................................. 134
6.4.4 The On-Board Real-Time Clock ............................................. 135

6.5 VMEMROM Commands ............................................................... 135
6.5.1 ARB – Set the Arbiter of the CPU Board .................................... 136
6.5.2 CONFIG – Search VMEbus for Hardware ................................... 136
6.5.3 FERASE – Erase Flash Memories ............................................. 137
6.5.4 FGA – Change Boot Setup for Gate Array ................................... 138
6.5.5 FLUSH – Set Buffered Write Mode .......................................... 139
6.5.6 FMB – FORCE Message Broadcast .......................................... 140
6.5.7 FPROG – Program Flash Memories ......................................... 141
6.5.8 FUNCTIONAL – Perform Functional Test ................................... 142
6.5.9 INFO – Information about the CPU Board .................................. 143
6.5.10 MEM – Set Data Bus Width of the VMEbus ............................. 143
6.5.11 RUNINRAM – Run VMEMROM in RAM .................................... 144
6.5.12 RUNINROM – Run VMEMROM in ROM ................................... 145
6.5.13 SELFTEST – Perform On-Board Selftest ................................... 145

6.6 Installing a New Hard Disk (Using FRMT and INIT) .............................. 146

7 Appendix to VMEMROM ................................................................. 151
7.1 Driver Installation ................................................................. 151
7.1.1 VMEbus Memory ................................................................. 151
7.1.2 SYS68K/SIO-1/2 ................................................................. 152
7.1.3 SYS68K/ISIO-1/2 ................................................................. 153
7.1.4 SYS68K/WFC-1 Disk Controller ............................................. 155
7.1.5 SYS68K/ISCSI-1 Disk Controller ........................................... 156
7.1.6 Local SCSI Controller ........................................................... 157

7.2 S-Record Formats ................................................................. 158

7.3 System RAM Definitions .............................................................. 159
7.4 Task Control Block Definitions ........................................... 162
7.5 Interrupt Vector Table of VMEPROM ................................... 165
7.6 Benchmark Source Code .................................................... 168
7.7 Modifying Special Locations in ROM .................................... 174
7.8 Binding Applications to VMEPROM ..................................... 177
  7.8.1 Using External Memory .............................................. 177
  7.8.2 Using System Flash Memory ......................................... 177

8 FGA Boot ............................................................................. 179
  8.1 Boot Sequence ............................................................. 179
  8.2 Debugger Commands ..................................................... 182
    8.2.1 AS – Line Assembler ............................................... 182
    8.2.2 BANNER – Display Banner Again ............................... 184
    8.2.3 CONT – Continue with Calling Routine ....................... 184
    8.2.4 DI – Disassembler .................................................. 184
    8.2.5 DRAMINIT – Initialize DRAM .................................... 185
    8.2.6 FERASE – Erase Flash Memories ............................... 185
    8.2.7 FPROG – Program Flash Memories .............................. 186
    8.2.8 GO – Go to Subroutine ............................................ 188
    8.2.9 LO – Load S-Records to Memory ................................ 188
    8.2.10 NETLOAD – Load File via Network to Memory ............. 190
    8.2.11 NETSAVE – Save Data via Network to File ................. 191
    8.2.12 SETUP – Change Initialization Values ....................... 193
    8.2.13 SLOT – Change Slot Number and VMEbus Slave Address . 194
    8.2.14 VMEADDR – Change VMEbus Slave Address ............... 195

  8.3 FGA Boot Utility Functions ............................................. 196
    8.3.1 Extended Flash Memory Programming ......................... 197
    8.3.2 Erase Flash Memories .............................................. 198
    8.3.3 Get System Values in SRAM ...................................... 199
    8.3.4 Get Application Values in SRAM ............................... 200
    8.3.5 Get Ethernet Number ............................................. 200
    8.3.6 Get Memory Limits ................................................ 201
# List of Tables and Figures

<table>
<thead>
<tr>
<th>Referenced manuals separately available from FORCE COMPUTERS</th>
<th>Page</th>
<th>Tab./Fig.</th>
</tr>
</thead>
<tbody>
<tr>
<td>History of manual publication</td>
<td>x</td>
<td>Tab. 1</td>
</tr>
<tr>
<td>Fonts, Notations and Conventions</td>
<td>xi</td>
<td>Tab. 2</td>
</tr>
<tr>
<td>Specification for the SYS68K/CPU-60 board</td>
<td>2</td>
<td>Tab. 1</td>
</tr>
<tr>
<td>Ordering information for the SYS68K/CPU-60</td>
<td>6</td>
<td>Tab. 2</td>
</tr>
<tr>
<td>Block diagram of the SYS68K/CPU-60</td>
<td>9</td>
<td>Fig. 1</td>
</tr>
<tr>
<td>Location diagram of the SYS68K/CPU-60 (schematic)</td>
<td>10</td>
<td>Fig. 2</td>
</tr>
<tr>
<td>Switch settings</td>
<td>11</td>
<td>Tab. 3</td>
</tr>
<tr>
<td>Front panel features</td>
<td>15</td>
<td>Tab. 4</td>
</tr>
<tr>
<td>Pinout of the front panel serial I/O ports config. for RS-232</td>
<td>19</td>
<td>Tab. 5</td>
</tr>
<tr>
<td>Pinout of the front panel serial I/O ports config. for RS-422</td>
<td>19</td>
<td>Tab. 6</td>
</tr>
<tr>
<td>Pinout of the front panel serial I/O ports config. for RS-485</td>
<td>20</td>
<td>Tab. 7</td>
</tr>
<tr>
<td>15-pin AUI-Ethernet connector</td>
<td>24</td>
<td>Tab. 8</td>
</tr>
<tr>
<td>P2 connector pinout with serial I/O config. for RS-232</td>
<td>25</td>
<td>Fig. 3</td>
</tr>
<tr>
<td>P2 connector pinout with serial I/O config. for RS-422</td>
<td>26</td>
<td>Fig. 4</td>
</tr>
<tr>
<td>P2 connector pinout with serial I/O config. for RS-485</td>
<td>27</td>
<td>Fig. 5</td>
</tr>
<tr>
<td>SYS68K/IOBP-1 pin assignment for VME P2</td>
<td>28</td>
<td>Fig. 6</td>
</tr>
<tr>
<td>POST codes indicating boot status</td>
<td>30</td>
<td>Tab. 9</td>
</tr>
<tr>
<td>SYS68K/CPU-60 block diagram</td>
<td>35</td>
<td>Fig. 7</td>
</tr>
<tr>
<td>SYS68K/CPU-60 memory map</td>
<td>37</td>
<td>Tab. 10</td>
</tr>
<tr>
<td>SYS68K/CPU-60 interrupt map</td>
<td>39</td>
<td>Tab. 11</td>
</tr>
<tr>
<td>SYS68K/CPU-60 parameters and timers register map and CIO loc.</td>
<td>41</td>
<td>Tab. 12</td>
</tr>
<tr>
<td>CIO1 port C data register</td>
<td>41</td>
<td>Tab. 13</td>
</tr>
<tr>
<td>CIO1 port B data register</td>
<td>42</td>
<td>Tab. 14</td>
</tr>
<tr>
<td>CIO1 port A data register (including MODE x status register)</td>
<td>43</td>
<td>Tab. 15</td>
</tr>
<tr>
<td>CIO2 port C data register</td>
<td>44</td>
<td>Tab. 16</td>
</tr>
<tr>
<td>CIO2 port B data register</td>
<td>44</td>
<td>Tab. 17</td>
</tr>
<tr>
<td>Naming the parts of the front-panel hexadecimal display</td>
<td>45</td>
<td>Fig. 8</td>
</tr>
<tr>
<td>CIO2 port A data register</td>
<td>45</td>
<td>Tab. 18</td>
</tr>
<tr>
<td>68060 CPU exception vector assignments</td>
<td>46</td>
<td>Tab. 19</td>
</tr>
<tr>
<td>Watchdog register map (superset of the memory controller register map)</td>
<td>49</td>
<td>Tab. 20</td>
</tr>
<tr>
<td>Watchdog retrigger register (WDR)</td>
<td>49</td>
<td>Tab. 21</td>
</tr>
<tr>
<td>RIALTO bus bridge register map</td>
<td>50</td>
<td>Tab. 22</td>
</tr>
<tr>
<td>Bridge configuration register (BCR)</td>
<td>50</td>
<td>Tab. 23</td>
</tr>
<tr>
<td>Snoop window definition in BCR</td>
<td>51</td>
<td>Tab. 24</td>
</tr>
<tr>
<td>Memory controller register map (included in the watchdog register map)</td>
<td>53</td>
<td>Tab. 25</td>
</tr>
<tr>
<td>MCR, memory configuration register</td>
<td>54</td>
<td>Tab. 26</td>
</tr>
<tr>
<td>Memory diagnostic register (MDR)</td>
<td>55</td>
<td>Tab. 27</td>
</tr>
<tr>
<td>DRAM capacity encoding at CIOx port C data registers</td>
<td>57</td>
<td>Tab. 28</td>
</tr>
</tbody>
</table>
### Tables and Figures

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>DRAM device types and number of used banks</td>
<td>58</td>
</tr>
<tr>
<td>Default DRAM access address ranges from the 68060 CPU</td>
<td>59</td>
</tr>
<tr>
<td>User SRAM features</td>
<td>61</td>
</tr>
<tr>
<td>System PROM features</td>
<td>62</td>
</tr>
<tr>
<td>System PROM device types</td>
<td>63</td>
</tr>
<tr>
<td>System PROM address map</td>
<td>63</td>
</tr>
<tr>
<td>Boot PROM features</td>
<td>65</td>
</tr>
<tr>
<td>Boot PROM address map, factory options, and device types</td>
<td>66</td>
</tr>
<tr>
<td>User flash factory options and device types</td>
<td>68</td>
</tr>
<tr>
<td>Local SRAM features</td>
<td>69</td>
</tr>
<tr>
<td>Local SRAM factory options and device types</td>
<td>70</td>
</tr>
<tr>
<td>RTC 72423 features</td>
<td>71</td>
</tr>
<tr>
<td>RTC registers address map</td>
<td>72</td>
</tr>
<tr>
<td>Address modifier (AM) ranges A32/A24/A16</td>
<td>77</td>
</tr>
<tr>
<td>Address ranges related to AM codes</td>
<td>77</td>
</tr>
<tr>
<td>Bus widths related to address ranges: VMEbus master interface</td>
<td>78</td>
</tr>
<tr>
<td>VMEbus master transfer cycles defined for D32 data bus width</td>
<td>79</td>
</tr>
<tr>
<td>VMEbus master transfer cycles defined for D16 data bus width</td>
<td>79</td>
</tr>
<tr>
<td>VMEbus slave AM codes</td>
<td>82</td>
</tr>
<tr>
<td>Valid configurations for VMEbus release modes</td>
<td>84</td>
</tr>
<tr>
<td>Slot-1 status register (RO)</td>
<td>88</td>
</tr>
<tr>
<td>Serial I/O channel register map and hybrid locations</td>
<td>90</td>
</tr>
<tr>
<td>Bit 7 of the WR5 register</td>
<td>90</td>
</tr>
<tr>
<td>SCSI-53C720SE GPCNTL register</td>
<td>91</td>
</tr>
<tr>
<td>SCSI-53C720SE GPREG register</td>
<td>91</td>
</tr>
<tr>
<td>FDC register map</td>
<td>95</td>
</tr>
<tr>
<td>Block diagram of the Ethernet interface</td>
<td>96</td>
</tr>
<tr>
<td>Initializing the LAN AM 79C965A register access</td>
<td>98</td>
</tr>
<tr>
<td>Ethernet controller address layout (16-Bit mode)</td>
<td>99</td>
</tr>
<tr>
<td>Example word-swapped init. block for LAN AM 79C965A in 16-bit mode</td>
<td>100</td>
</tr>
<tr>
<td>Initializing a receive descriptor in 16-bit mode</td>
<td>100</td>
</tr>
<tr>
<td>Examples for power-up and reset configuration by rotary switches</td>
<td>129</td>
</tr>
<tr>
<td>Power-up and reset actions defined by rotary switch MODE 2</td>
<td>130</td>
</tr>
<tr>
<td>Power-up and reset actions defined by rotary switch MODE 1</td>
<td>131</td>
</tr>
<tr>
<td>Main memory layout</td>
<td>132</td>
</tr>
<tr>
<td>Layout of system flash memory</td>
<td>133</td>
</tr>
<tr>
<td>On-board I/O devices</td>
<td>133</td>
</tr>
<tr>
<td>On-board interrupt sources</td>
<td>134</td>
</tr>
<tr>
<td>Off-board interrupt sources</td>
<td>135</td>
</tr>
<tr>
<td>Base addresses of SIO-1/2 ports</td>
<td>153</td>
</tr>
<tr>
<td>Base addresses of ISIO-1/2 ports</td>
<td>154</td>
</tr>
<tr>
<td>Types of S-record format modules and VMEPROM support</td>
<td>158</td>
</tr>
<tr>
<td>User patch table</td>
<td>175</td>
</tr>
<tr>
<td>Boot up procedure</td>
<td>180</td>
</tr>
<tr>
<td>Boot up procedure (continued)</td>
<td>181</td>
</tr>
<tr>
<td>Stack frame</td>
<td>196</td>
</tr>
</tbody>
</table>
Using This Manual

This section does not provide information on the product but on common features of the manual itself:

- its structure,
- special layout conventions,
- and related documents.

Audience of the Manual

This Technical Reference Manual is intended for hard- and software developers installing and integrating the SYS68K/CPU-60 into their systems.

Overview of the Manual

This Technical Reference Manual provides a comprehensive hardware and software guide to your board.

IMPORTANT Please take a moment to examine the “Table of Contents” to see how this documentation is structured. This will be of value to you when looking for information in the future.

It includes

- a brief overview of the product, the specifications, the ordering information: see section 1 “Introduction” on page 1.
- the installation instructions for powering up the board: see section 2 “Installation” on page 7. It includes the default configuration (switches and the like), initialization, and connector pinouts.

The installation instructions also appear as the product’s installation guide – a separate manual delivered together with each product shipped.

- a detailed hardware description: see section 3 “Hardware” on page 33.
- the circuit schematics of the board for reference purposes.

The circuit schematics are packaged separately to enable easy updating. They will always be shipped together with this manual. Therefore:

Insert the circuit schematics now: see section 4 “Circuit Schematics”.
• the data sheets of board components that are relevant for configuring and integrating the board in systems. The following data sheets are delivered:
  – Motorola 68060 (delivered as a separate manual)
  – CIO Z8536
  – FDC 37C65C: pin-to-pin compatible with industry standard WD37C65C
  – LAN AM 79C965A
  – RTC 72421
  – SCC AM 85C30
  – SCSI 53C720SE

The data sheets are packaged separately to enable easy updating. They are always shipped together with this manual. Therefore:

☞ Insert the data sheets now: see section 5 “Data Sheets”.

• a detailed description of VMEPROM and FGA Boot which control the CPU board operations: see section 6 “VMEPROM” on page 125, section 7 “Appendix to VMEPROM” on page 151, and section 8 “FGA Boot” on page 179.

There is additional space allocated in the manual for user notes, modifications, etc.

**Referenced Manuals**

<table>
<thead>
<tr>
<th>Table 1 Referenced manuals separately available from FORCE COMPUTERS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Referenced manual</td>
</tr>
<tr>
<td>----------------------------------</td>
</tr>
<tr>
<td>IOPI-2 User’s Installation</td>
</tr>
<tr>
<td>SIO User's Manual</td>
</tr>
</tbody>
</table>
Publication History of the Manual

Table 2

<table>
<thead>
<tr>
<th>Edition</th>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>July 1996</td>
<td>First print</td>
</tr>
<tr>
<td>2</td>
<td>August 1996</td>
<td>Editorial Changes</td>
</tr>
<tr>
<td>2.1</td>
<td>July 1997</td>
<td>Extended NETLOAD and NETSAVE FGA Boot debugger commands. Corrected FDC register map, corrected pinout of front-panel serial I/O port, and corrected units for VME- BUSTIMER bits (memory configuration register).</td>
</tr>
<tr>
<td>2.2</td>
<td>January 1999</td>
<td>Switch settings for RS-485 corrected, RS-485 configuration described</td>
</tr>
</tbody>
</table>

Fonts, Notations and Conventions

Table 3

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000.0000 16</td>
<td>Typical notation for hexadecimal numbers (digits are 0 through F), e.g. used for addresses and offsets. Note the dot marking the 4th (to its right) and 5th (to its left) digit.</td>
</tr>
<tr>
<td>0000 8</td>
<td>Same for octal numbers (digits are 0 through 7)</td>
</tr>
<tr>
<td>0000 2</td>
<td>Same for binary numbers (digits are 0 and 1)</td>
</tr>
<tr>
<td>Program</td>
<td>Typical character format used for names, values, and the like that should be used typing literally the same word. Also used for on-screen-output.</td>
</tr>
<tr>
<td>Variable</td>
<td>Typical character format for words that represent a part of a command, a programming statement, or the like and that will be replaced by an applicable value when actually applied.</td>
</tr>
</tbody>
</table>
Icons for Ease of Use: Safety Notes and Tips & Tricks

There are 3 levels of safety notes used in this manual which are described below in brief by displaying a typical layout example.

Be sure to always read and follow the safety notes of a section first – before acting as documented in the other parts of the section.

CAUTION

Dangerous situation: injuries of people and severe damage to objects possible.

NOTICE

Possibly dangerous situation: no injuries to people but damage to objects possible.

IMPORTANT

No danger encountered. Only application hints and time-saving tips & tricks or information on typical errors when using the information mentioned below this safety hint.
1 Introduction

The SYS68K/CPU-60 is a high performance single-board computer providing an A32/D32 VMEbus interface including DMA. It is based on
- the 68060 CPU,
- the FORCE gate array FGA-002,
- and the VMEbus.

Memory

The SYS68K/CPU-60 provides up to 32 Mbyte DRAM on-board (field upgradable). Up to 128 Mbyte DRAM are available with the SYS68K/MEM-60 extension module. Up to 2 Mbyte user SRAM, up to 512 Kbyte local SRAM with battery backup, up to 8 Mbyte system PROM, and up to 1 Mbyte boot PROM are available.

The shared DRAM is accessible from the 68060 CPU, the FGA-002 on-chip DMA controller, the SCSI on-chip DMA controller, the LAN on-chip DMA controller, and also from VMEbus masters.

Interfaces

The SYS68K/CPU-60 incorporates SCSI-2, Ethernet, and serial I/O on-board to provide full single-board computer functionality.

The SYS68K/CPU-60 has 2 serial ports at the front panel permitting a console port for download and data communication. Both ports use standard 9-pin D-Sub connectors.

CPU speed

The 68060 CPU runs at 50 MHz and has cache snooping support for alternate master access to the shared DRAM.

Real-time clock

A real-time clock with battery backup is also available.
1.1 Specification

<table>
<thead>
<tr>
<th>Table 1</th>
<th>Specification for the SYS68K/CPU-60 board</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU type</td>
<td>With snooping support (write through for shared data necessary)</td>
</tr>
<tr>
<td>CPU clock frequency</td>
<td></td>
</tr>
<tr>
<td>CPU bus frequency (half of CPU clock frequency)</td>
<td></td>
</tr>
<tr>
<td>DRAM</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CPU-60D/4</td>
</tr>
<tr>
<td></td>
<td>CPU-60D/8</td>
</tr>
<tr>
<td></td>
<td>(Upgradable with MEM-60/8 to 16 Mbyte in total)</td>
</tr>
<tr>
<td></td>
<td>CPU-60D/16</td>
</tr>
<tr>
<td></td>
<td>(Upgradable with MEM-60/16 to 32 Mbyte in total)</td>
</tr>
<tr>
<td></td>
<td>CPU-60D/32</td>
</tr>
<tr>
<td></td>
<td>32-bit wide, byte parity, accessible from the CPU, FGA-002, SCSI and Ethernet on-chip DMA controller, and also from other VMEbus masters</td>
</tr>
<tr>
<td>System PROM</td>
<td>Flash memory, 32-bit wide memory data path, reprogrammable on-board, hardware write protection (independent from boot PROM)</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>User SRAM</td>
<td>32-bit wide, with on-board battery and +5VSTDBY line backup, accessible from the CPU, SCSI and Ethernet on-chip DMA controller, and also from other VMEbus masters</td>
</tr>
<tr>
<td>Boot PROM</td>
<td>12V flash memory, 8-bit wide, reprogrammable on-board in case of flash memory, hardware write protection in case of flash memory (independent from system PROM), 32-pin PLCC sockets</td>
</tr>
<tr>
<td></td>
<td>Factory options:</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(more configurations possible)</td>
</tr>
</tbody>
</table>
### Table 1 Specification for the SYS68K/CPU-60 board (cont.)

<table>
<thead>
<tr>
<th>Feature</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local SRAM</td>
<td>8-bit wide, with on-board battery and +5VSTDBY line backup</td>
</tr>
<tr>
<td></td>
<td>32 Kbyte (factory option)</td>
</tr>
<tr>
<td></td>
<td>128 Kbyte</td>
</tr>
<tr>
<td></td>
<td>512 Kbyte (factory option)</td>
</tr>
<tr>
<td>User flash</td>
<td>8-bit wide, reprogrammable on-board, hardware write protectable</td>
</tr>
<tr>
<td></td>
<td>128 Kbyte (factory option)</td>
</tr>
<tr>
<td></td>
<td>256 Kbyte</td>
</tr>
<tr>
<td></td>
<td>512 Kbyte (factory option)</td>
</tr>
<tr>
<td>Serial I/O interfaces</td>
<td>available via the front panel (permitting a console port, download, and data communication)</td>
</tr>
<tr>
<td></td>
<td>available via the 3-row VME P2 connector</td>
</tr>
<tr>
<td></td>
<td>RS-232, RS-422, or RS-485 compatible via FORCE hybrids FH-00x, SDLC, HDLC, IBM BISYNC, and ASYNC protocol support, up to 38.4 Kbit/s asynchronous data rate</td>
</tr>
<tr>
<td></td>
<td>SCC 85C30</td>
</tr>
<tr>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>1 (2 as factory option, the second port is not available with 16-bit wide SCSI option)</td>
</tr>
<tr>
<td>Ethernet interface on front panel</td>
<td>Via AM 79C965</td>
</tr>
<tr>
<td></td>
<td>AUI via 15-pin D-Sub</td>
</tr>
<tr>
<td></td>
<td>Cheapernet via SMB (factory option)</td>
</tr>
<tr>
<td>SCSI interface, single-ended</td>
<td>Via NCR 53C720SE</td>
</tr>
<tr>
<td></td>
<td>8-bit (fast)</td>
</tr>
<tr>
<td></td>
<td>16-bit (wide; factory option, the wide SCSI option is not available with 2 serial ports on the P2 connector)</td>
</tr>
<tr>
<td>Floppy disk interface</td>
<td>FDC 37C65C</td>
</tr>
<tr>
<td>SYS68K/CPU-60 parameters controllable</td>
<td>Via CIO Z8536</td>
</tr>
<tr>
<td>Timers</td>
<td>Via 2 CIO Z8536</td>
</tr>
<tr>
<td></td>
<td>Six 16-bit timers</td>
</tr>
<tr>
<td>Watchdog timer</td>
<td>Reset/NMI</td>
</tr>
<tr>
<td>Real-time clock</td>
<td>With on-board battery and +5VSTDBY line backup; IRQ capability</td>
</tr>
<tr>
<td></td>
<td>Via RTC 72423</td>
</tr>
</tbody>
</table>
Table 1 Specification for the SYS68K/CPU-60 board (cont.)

<table>
<thead>
<tr>
<th>VMEbus interface</th>
<th>Via FGA-002</th>
</tr>
</thead>
<tbody>
<tr>
<td>Master</td>
<td>A32, A24, A16: D8, D16, D32, UAT, RMW</td>
</tr>
<tr>
<td>Slave</td>
<td>A32, A24: D8, D16, D32, UAT, RMW</td>
</tr>
<tr>
<td>Slave AM CODES:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Standard supervisory data/program access</td>
</tr>
<tr>
<td></td>
<td>Standard non-privileged data/program access</td>
</tr>
<tr>
<td></td>
<td>Short supervisory access</td>
</tr>
<tr>
<td></td>
<td>Short non-privileged access</td>
</tr>
<tr>
<td></td>
<td>Extended supervisory data/program access</td>
</tr>
<tr>
<td></td>
<td>Extended non-privileged data/program access</td>
</tr>
<tr>
<td>Arbiter</td>
<td>Single-level with arbitration timeout</td>
</tr>
<tr>
<td>Arbiter request modes</td>
<td>ROR, RBCLR, REC, RAT yes</td>
</tr>
<tr>
<td>SYSCLK driver</td>
<td>yes</td>
</tr>
<tr>
<td>IACK daisy chain driver</td>
<td>yes</td>
</tr>
<tr>
<td>Slot 1 function switch</td>
<td>yes</td>
</tr>
<tr>
<td>Mailbox interrupts</td>
<td>8</td>
</tr>
<tr>
<td>FORCE Message Broadcast</td>
<td>8 Byte</td>
</tr>
<tr>
<td>FMB-FIFO 0</td>
<td>1 Byte</td>
</tr>
<tr>
<td>FMB-FIFO 1</td>
<td></td>
</tr>
<tr>
<td>Interrupts</td>
<td></td>
</tr>
<tr>
<td>VMEbus and local interrupt handler</td>
<td>1 to 7</td>
</tr>
<tr>
<td>Programmable IRQ levels for all sources</td>
<td>yes</td>
</tr>
<tr>
<td>Total number of IRQ sources</td>
<td>42</td>
</tr>
<tr>
<td>Reset and abort switches</td>
<td>yes</td>
</tr>
<tr>
<td>VMEPROM firmware installed on all board versions</td>
<td>512 Kbyte</td>
</tr>
<tr>
<td><strong>Table 1</strong> Specification for the SYS68K/CPU-60 board  (cont.)</td>
<td></td>
</tr>
<tr>
<td>-------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td><strong>Power requirements (for a SYS68K/CPU-60D/32)</strong></td>
<td>3.5 A typical</td>
</tr>
<tr>
<td>+ 5 V max</td>
<td>0.1 A typical – with no</td>
</tr>
<tr>
<td>+12 V max</td>
<td>Ethernet MAU</td>
</tr>
<tr>
<td>- 12 V max</td>
<td>plugged</td>
</tr>
<tr>
<td></td>
<td>0.1 A typical</td>
</tr>
<tr>
<td><strong>Backup battery at location BAT 1</strong></td>
<td>CR2032-type lithium battery</td>
</tr>
<tr>
<td><strong>Front panel features</strong></td>
<td>Reset and abort key</td>
</tr>
<tr>
<td></td>
<td>4 Status LEDs</td>
</tr>
<tr>
<td></td>
<td>7-segment display</td>
</tr>
<tr>
<td></td>
<td>2 rotary switches</td>
</tr>
<tr>
<td><strong>Operating temperature with forced air cooling</strong></td>
<td>0 °C to +55 °C</td>
</tr>
<tr>
<td><strong>Storage temperature</strong></td>
<td>-40 °C to +85 °C</td>
</tr>
<tr>
<td>Without battery</td>
<td>+60 °C</td>
</tr>
<tr>
<td>With installed battery</td>
<td>5 % to 95 %</td>
</tr>
<tr>
<td><strong>Relative humidity (non-condensing)</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Board dimensions</strong></td>
<td>160 mm x 233 mm</td>
</tr>
<tr>
<td><strong>No. of slots used</strong></td>
<td>1</td>
</tr>
<tr>
<td><strong>Standards compliance</strong></td>
<td>ANSI/VITA 1-1994</td>
</tr>
<tr>
<td>VMEbus interface</td>
<td></td>
</tr>
</tbody>
</table>
### 1.2 Ordering Information

#### Table 2
Ordering information for the SYS68K/CPU-60

<table>
<thead>
<tr>
<th>Product name</th>
<th>Product description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYS68K/CPU-60D/4, .../8, .../16, .../32 MEM-60/8, .../16</td>
<td>50 MHz 68060 based CPU board (“60D” in product name, with “60E” it is based on a 66 MHz 68060), 32-bit DMA, 4 (8/16/32) Mbyte shared memory, 2 serial I/O channels (RS-232), SCSI, floppy disk and Ethernet interface, VMEPROM documentation not included. SYS68K/CPU-60D/8 field upgradable to a total of 16 Mbyte shared memory by installing the MEM-60/8 memory module (SYS68K/CPU-60x/16 field upgradable to a total of 32 Mbyte by MEM-60/16).</td>
</tr>
<tr>
<td>SYS68K/CPU-60Lite/4</td>
<td>50 MHz 68LC060 based CPU board, 32-bit DMA, 4 (8/16/32) Mbyte shared memory, 2 serial I/O channels (RS-232), VMEPROM documentation not included.</td>
</tr>
<tr>
<td>UM SYS68K/VMEPROM/32</td>
<td>VMEPROM User's Manual for 32-bit CPUs</td>
</tr>
<tr>
<td>SYS68K/IOBP-1</td>
<td>Rear I/O paddel panel for single board computers providing connectors for 8-bit SCSI, floppy disk drive and one serial I/O port.</td>
</tr>
<tr>
<td>IOPI-2</td>
<td>Rear I/O paddel panel for single board computers providing connectors for 8-bit SCSI, floppy disk drive and one serial I/O port.</td>
</tr>
<tr>
<td>SYS68K/CABLE 9-25 SET</td>
<td>Set of 4 adapter cables 9-pin D-Sub male connector to 25-pin D-Sub female connector, length 2 m (SYS68K/CPU-60)</td>
</tr>
<tr>
<td>SYS68K/FH002/SET</td>
<td>Hybrids for the serial I/O interfaces (10 hybrids per set): RS-232 protocol</td>
</tr>
<tr>
<td>SYS68K/FH003/SET</td>
<td>RS-422 protocol</td>
</tr>
<tr>
<td>SYS68K/FH007/SET</td>
<td>RS-485 protocol</td>
</tr>
</tbody>
</table>
2 Installation

2.1 Safety Note

To ensure proper functioning of the product during its usual lifetime, take the following precautions before handling the board.

**CAUTION**
Malfunction or damage to the board or connected components
Electrostatic discharge and incorrect board installation and uninstallation can damage circuits or shorten their lifetime.

- Before installing or uninstalling the board, read this *Installation* section.
- Before installing or uninstalling MEM-60 memory modules, read the *MEM-60 Installation Guide* packaged together with the modules.
- Before installing or uninstalling the board, in a VME rack turn off the power.
- Before touching integrated circuits, ensure that you are working in an electrostatic free environment.
- Ensure that the board is connected to the VMEbus via both the P1 and the P2 connectors and that power is available on both of them.
- When operating the board in areas of strong electromagnetic radiation, ensure that the board
  - is bolted on the VME rack
  - and shielded by closed housing.

**CAUTION**
Damage of components caused by inappropriate floppy drive installation
There are floppy disk drives that provide means to connect the floppy disk drive frame electrically with DC ground, e.g., by inserting a jumper on the floppy disk drive.

- Before installing a floppy disk drive, always make sure that the floppy disk drive’s frame is not electrically connected with DC ground.

**NOTICE**
Damaging SYS68K/CPU-60 components
On the backplane the jumper for IACKIN-IACKOUT-bypass must be removed for proper operation. This is not necessary on active backplanes.
CAUTION

Maintenance of the CPU board:

The board is designed to be maintenance-free. However, note that a Lithium battery is installed on the board. The battery provides a data retention of 7 years summing up all periods of actual battery use. Therefore, FORCE COMPUTERS assumes that there usually is no need to exchange the Lithium battery except for example in the case of long-term spare part handling. Observe the following safety notes:

• Incorrect exchange of Lithium batteries can result in a hazardous explosion.

• Exchange the battery before 7 years of actual battery use have elapsed.

• Exchanging the battery always results in data loss of the devices which use the battery as power backup. Therefore, back up affected data before exchanging the battery.

• Always use the same type of Lithium battery as is already installed.

• When installing the new battery ensure that the marked dot on top of the battery covers the dot marked on the chip.

• Used batteries have to be disposed according to your country’s legislation.

2.2 Installation Prerequisites and Requirements

IMPORTANT

Before powering up

• check this section for installation prerequisites and requirements

• and check the consistency of the current switch settings (see section 2.4 “Switch Settings” on page 11).

2.2.1 Requirements

The installation requires only

• a power supply

• and a VMEbus backplane with P1 and P2 connector.

Power supply

The power supply must meet the following specifications:

• required for the processor board: +5 V (3.5 A typical for a SYS68K/CPU-60D/32, 3.0 A typical for a SYS68K/CPU-60D/4)

• required for the RS-232 serial interface and the Ethernet interface:
  – +12 V (0.1 A typical – with no Ethernet MAU plugged)
  – and –12 V (0.1 A typical)
2.2.2 Terminal Connection

For the initial power-up, a terminal can be connected to the standard 9-pin D-Sub connector of serial port 1, which is located at the front panel (see section 2.7 “Serial I/O Ports – SCC” on page 17).

2.2.3 Functional and Location Overview

Figure 1 gives a functional overview, figure 2 highlights the locations of the important SYS68K/CPU-60 components.

**Figure 1** Block diagram of the SYS68K/CPU-60
Figure 2  Location diagram of the SYS68K/CPU-60 (schematic)
2.3 Automatic Power Up – Voltage Sensor and Watchdog Timer

In the following situations the CPU board will automatically be reset and proceed with a normal power up:

Voltage sensor • The voltage sensor generates a reset when the voltage level drops below 4.75 V.

Watchdog timer • Per factory default the watchdog timer is disabled. If the watchdog timer is enabled, it generates a non-maskable interrupt (NMI) followed by a pseudo power up when it is not re-triggered. The watchdog timer can be enabled by software.

2.4 Switch Settings

The following table lists the function and the default settings of all switches shown in figure 2 “Location diagram of the SYS68K/CPU-60 (schematic)” on page 10.

IMPORTANT • Before powering up the board check the current switch settings for consistency.

• SW6-1, SW6-2, SW6-3, and SW6-4 will only be read on a power up.

<table>
<thead>
<tr>
<th>Name and default setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SW5-1</td>
<td>On-board power backup from VME standby</td>
</tr>
<tr>
<td></td>
<td>OFF = disabled</td>
</tr>
<tr>
<td></td>
<td>ON = enabled</td>
</tr>
<tr>
<td>SW5-2</td>
<td>On-board power backup from backup battery</td>
</tr>
<tr>
<td></td>
<td>OFF = disabled</td>
</tr>
<tr>
<td></td>
<td>ON = enabled</td>
</tr>
<tr>
<td>SW5-3</td>
<td>Devices with backup</td>
</tr>
<tr>
<td></td>
<td>OFF = RTC</td>
</tr>
<tr>
<td></td>
<td>ON = RTC, local and user SRAM</td>
</tr>
<tr>
<td>SW5-4</td>
<td>reserved: must be OFF</td>
</tr>
<tr>
<td>Name and default setting</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| SW6-1 OFF               | Slot 1 auto-detection  
|                          | OFF = enabled  
|                          | ON = disabled (also called manual mode) |
| SW6-2 OFF               | Slot 1 manual mode: only available when SW6-1 = ON  
|                          | OFF = disabled  
|                          | ON = enabled |
| SW6-3 OFF               | VMEbus arbitration level (BRx* signals)  
| SW6-3 OFF               | SW6-4 Level  
| OFF                      | OFF = level 3 (BR3*)  
| ON                       | OFF = level 2 (BR2*)  
| ON                       | OFF = level 1 (BR1*)  
| ON                       | ON = level 0 (BR0*) |
| SW7-1 OFF               | Boot PROM configuration  
|                          | OFF = Socket 1 – 0…512 Kbyte,  
|                          | ON = Socket 1 disabled,  
|                          | Socket 2 – 512 Kbyte…1 Mbyte  
|                          | ON = Socket 2 from 0…1 Mbyte |
| SW7-2 OFF               | Abort key  
|                          | OFF = enabled  
|                          | ON = disabled |
| SW7-3 OFF               | Reset key  
|                          | OFF = enabled  
|                          | ON = disabled |
| SW7-4 OFF               | Boot PROM write protection  
|                          | OFF = write-protected  
|                          | ON = writing enabled |
### Switch settings (cont.)

<table>
<thead>
<tr>
<th>Name and default setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SW9-1 OFF</td>
<td>Power up detection level&lt;br&gt;OFF = conforms to VME specification&lt;br&gt;ON = below VME specification</td>
</tr>
<tr>
<td>SW9-2 OFF</td>
<td>The switch setting signals to software:&lt;br&gt;DRAM parity check should be&lt;br&gt;OFF = enabled&lt;br&gt;ON = disabled</td>
</tr>
<tr>
<td>SW9-3 OFF</td>
<td>VMEbus SYSRESET output&lt;br&gt;OFF = enabled&lt;br&gt;ON = disabled</td>
</tr>
<tr>
<td>SW9-4 OFF</td>
<td>VMEbus SYSRESET input&lt;br&gt;OFF = enabled&lt;br&gt;ON = disabled</td>
</tr>
<tr>
<td>SW10-1 OFF</td>
<td>Configuration of serial port 2 depending on SW10-1, SW12-2, and SW12-3&lt;br&gt;Switch Configuration&lt;br&gt;10-1 12-2 12-3&lt;br&gt;OFF OFF OFF = RS-232 async.&lt;br&gt;ON ON OFF = RS-232 sync. slave&lt;br&gt;OFF OFF ON = RS-232 sync. master&lt;br&gt;ON ON ON = RS-422&lt;br&gt;ON ON OFF = RS-485</td>
</tr>
<tr>
<td>SW10-2 OFF</td>
<td>Configuration of serial port 1 depending on SW10-2, SW12-1, and SW12-4&lt;br&gt;Switch Configuration&lt;br&gt;10-2 12-1 12-4&lt;br&gt;OFF OFF OFF = RS-232 async.&lt;br&gt;ON OFF ON = RS-232 sync. slave&lt;br&gt;OFF ON OFF = RS-232 sync. master&lt;br&gt;ON ON ON = RS-422&lt;br&gt;ON OFF ON = RS-485</td>
</tr>
<tr>
<td>SW10-3 OFF</td>
<td>System PROM write protection&lt;br&gt;OFF = writing enabled&lt;br&gt;ON = write-protected</td>
</tr>
<tr>
<td>SW10-4 OFF</td>
<td>User flash write protection&lt;br&gt;OFF = writing enabled&lt;br&gt;ON = write-protected</td>
</tr>
<tr>
<td>Name and default setting</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>SW11-1 OFF</td>
<td>SCSI-termination</td>
</tr>
<tr>
<td>SW11-2 OFF</td>
<td>SCSI-termination for SW11-1 SW11-2</td>
</tr>
<tr>
<td></td>
<td>OFF OFF = wide and 8-bit SCSI</td>
</tr>
<tr>
<td></td>
<td>OFF ON = only upper 8 bits of wide SCSI</td>
</tr>
<tr>
<td></td>
<td>ON OFF = only 8-bit SCSI</td>
</tr>
<tr>
<td></td>
<td>ON ON = none</td>
</tr>
<tr>
<td>SW11-3 OFF</td>
<td>reserved: must be OFF.</td>
</tr>
<tr>
<td>SW11-4 OFF</td>
<td>reserved: must be OFF.</td>
</tr>
<tr>
<td>SW12-1 OFF</td>
<td>Configuration of serial port 1 depending on SW10-2, SW12-1, and SW12-4 (see SW10-2)</td>
</tr>
<tr>
<td>SW12-2 OFF</td>
<td>Configuration of serial port 2 depending on SW10-1, SW12-2, and SW12-3 (see SW10-1)</td>
</tr>
<tr>
<td>SW12-3 OFF</td>
<td></td>
</tr>
<tr>
<td>SW12-4 OFF</td>
<td>Configuration of serial port 1 depending on SW10-2, SW12-1, and SW12-4 (see SW10-2)</td>
</tr>
</tbody>
</table>
2.5 Front Panel

The features of the front panel are described in the following table. For a location diagram see figure 2 “Location diagram of the SYS68K/CPU-60 (schematic)” on page 10.

**IMPORTANT**

Toggling the reset key and the abort key at the same time has a special function which is described in the boot software description of the *FORCE Gate Array FGA-002 User's Manual*.

Table 4 Front panel features

<table>
<thead>
<tr>
<th>Device</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RESET</td>
<td>Mechanical reset key: When enabled and toggled it instantaneously affects the CPU board by generating a reset. Depending on SW9-3 the reset generates a VMEbus SYSRESET (see “SW9-3” on page 13). A reset of all on-board I/O devices and the CPU is performed when the reset key is pushed to the UP position. RESET is held active until the key is back in the DOWN position but at least 200 ms guaranteed by a local timer. Power fail (below approximately 4.7 Volts) and power up – both lasting at minimum 200 ms to 300 ms – also force a reset to start the CPU board. For information on enabling the key, see “SW7-3” on page 12.</td>
</tr>
<tr>
<td>ABORT</td>
<td>Mechanical abort key: When enabled and toggled it instantaneously affects the CPU board by generating an interrupt request (IRQ) on level 7 via the FGA-002. The abort key is activated in UP position and deactivated in DOWN position. This allows to implement an abort of the current program, to trigger a self-test or to start a maintenance program. For information on enabling the key, see “SW7-2” on page 12.</td>
</tr>
<tr>
<td>DIAG</td>
<td>Software programmable hexadecimal display for diagnostics: It can be accessed via the CIO2 port B data register.</td>
</tr>
<tr>
<td>Device</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>MODE 1</td>
<td>2 hexadecimal rotary switches, each decoded with 4 bit. The status of the rotary switch can be read in the CIO1 port A data register (including MODE x status register). Default for both rotary switches: F_{16}</td>
</tr>
<tr>
<td>RUN</td>
<td>68060 CPU status:</td>
</tr>
<tr>
<td></td>
<td>green</td>
</tr>
<tr>
<td></td>
<td>normal operation</td>
</tr>
<tr>
<td></td>
<td>red</td>
</tr>
<tr>
<td></td>
<td>the processor is halted or reset is active</td>
</tr>
<tr>
<td>BM</td>
<td>VME busmaster LED:</td>
</tr>
<tr>
<td></td>
<td>green</td>
</tr>
<tr>
<td></td>
<td>if the CPU board accesses the VMEbus as VMEbus master</td>
</tr>
<tr>
<td></td>
<td>off</td>
</tr>
<tr>
<td></td>
<td>otherwise</td>
</tr>
<tr>
<td>SYSF</td>
<td>SYSFAIL LED:</td>
</tr>
<tr>
<td></td>
<td>red</td>
</tr>
<tr>
<td></td>
<td>if SYSFAIL is asserted from the FGA-002</td>
</tr>
<tr>
<td></td>
<td>off</td>
</tr>
<tr>
<td></td>
<td>otherwise</td>
</tr>
<tr>
<td>UL</td>
<td>User LED: Software programmable by the RIALTO Bridge configuration register (BCR). Possible status:</td>
</tr>
<tr>
<td></td>
<td>green or off.</td>
</tr>
<tr>
<td>SERIAL 1</td>
<td>2 standard 9-pin D-Sub connectors for serial interface</td>
</tr>
<tr>
<td>SERIAL 2</td>
<td>(see section 2.7 “Serial I/O Ports – SCC” on page 17)</td>
</tr>
<tr>
<td>ETHERNET</td>
<td>15-pin AUI Ethernet connector for thick-wire Ethernet (802.3/10base5, see section 2.10 “Ethernet – LAN” on page 23); as factory option Cheapernet (802.3/10base2) is available via an SMB connector instead of the Ethernet AUI interface. An adapter from SMB type to BNC type connector is available from FORCE COMPUTERS.</td>
</tr>
</tbody>
</table>
2.6 SYS68K/CPU-60 Parameters and 16-bit Timers – CIO

<table>
<thead>
<tr>
<th>Devices: 2 CIO Z8536</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency</td>
<td>4 MHz</td>
</tr>
<tr>
<td>Package</td>
<td>44-pin PLCC</td>
</tr>
<tr>
<td>Accessible from</td>
<td>68060 CPU</td>
</tr>
<tr>
<td>Access address</td>
<td></td>
</tr>
<tr>
<td>for device #1</td>
<td>FF80.0C0016</td>
</tr>
<tr>
<td>for device #2</td>
<td>FF80.0E0016</td>
</tr>
<tr>
<td>Port width</td>
<td>Byte</td>
</tr>
<tr>
<td>Interrupt request level</td>
<td>Software programmable</td>
</tr>
<tr>
<td>FGA-002 interrupt</td>
<td>Local IRQ #4</td>
</tr>
</tbody>
</table>

Via the two CIO Z8536 devices several parameters can be configured or read, respectively: front panel rotary switch setting, front panel status display, on-board and MEM-60 DRAM size code, CPU-board code, availability of VME A24 extension, AUX DMA direction, programming voltage V_{PP}, configuration of FDC 37C65C control signals, ID-ROM (serial EEPROM), and the six 16-bit timers.

Timers

Six 16-bit timers with a resolution of 500 ns are available.

2.7 Serial I/O Ports – SCC

<table>
<thead>
<tr>
<th>Device: SCC AM 85C30</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency</td>
<td>8 MHz, 14.7456 MHz</td>
</tr>
<tr>
<td>Package</td>
<td>44-pin PLCC</td>
</tr>
<tr>
<td>Accessible from</td>
<td>68060 CPU</td>
</tr>
<tr>
<td>Access address</td>
<td>FF80.200016</td>
</tr>
<tr>
<td>Port width</td>
<td>Byte</td>
</tr>
<tr>
<td>Interrupt request level</td>
<td>Software programmable</td>
</tr>
<tr>
<td>FGA-002 interrupt</td>
<td>Local IRQ #5</td>
</tr>
</tbody>
</table>
The two serial I/O ports are available via 9-pin standard D-Sub connectors at the front panel. The SERIAL 1 front-panel port is also available on the VMEbus P2 connector (see section 2.11 “VMEbus P2 Connector Pinout” on page 24). All ports may be configured for RS-232, RS-422, and RS-485 standard conformance via installing the respective FORCE COMPUTERS hybrids FH-00x.

**Factory option**

As factory option the SERIAL 2 front-panel port is also available on the VMEbus P2 connector (see section 2.11 “VMEbus P2 Connector Pinout” on page 24). The SERIAL-2-on-P2 and the wide-SCSI factory option are not available simultaneously.

**Jumpers and terminations**

There are no on-board jumpers to configure the serial ports and no line terminations for RS-422 and RS-485 interfaces. If termination resistors are required to compensate various cable lengths and to reduce signal reflections, they must be installed externally to the SYS68K/CPU-60 (e.g. via a cable connector). The resistor value is application dependent, but a recommended value is 1000 Ω.

**Connector availability**

Both serial I/O ports 1 and 2 are available via a front-panel 9-pin D-Sub connector, per factory default only serial I/O port 1 is available via the P2 connector:

- serial I/O port 1 is wired to the front-panel connector labeled SERIAL 1 and to the VMEbus P2 connector with 7 lines,
- serial I/O port 2 is wired to the front-panel connector labeled SERIAL 2. As a factory option, serial I/O port 2 may also be wired to the VMEbus P2 connector (not available together with wide-SCSI factory option).

For the connection to the IOBP-1 back panel, see section 2.12 “SYS68K/IOBP-1” on page 28.

**Pinout**

For the front-panel pinout of the serial lines, see below. For the P2 pinout see section 2.11 “VMEbus P2 Connector Pinout” on page 24.
### Table 5: Pinout of the front panel serial I/O ports config. for RS-232

<table>
<thead>
<tr>
<th>Pin</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DCD (Data Carrier Detect, input)</td>
</tr>
<tr>
<td>2</td>
<td>RXD (Receive Data, input and output)</td>
</tr>
<tr>
<td>3</td>
<td>TXD (Transmit Data, output)</td>
</tr>
<tr>
<td>4</td>
<td>DTR (Data Terminal Ready, output)</td>
</tr>
<tr>
<td>5</td>
<td>GND (Ground)</td>
</tr>
<tr>
<td>6</td>
<td>DSR (Data Set Ready, input and output)</td>
</tr>
<tr>
<td>7</td>
<td>RTS (Request to Send, output)</td>
</tr>
<tr>
<td>8</td>
<td>CTS (Clear to Send, input)</td>
</tr>
<tr>
<td>9</td>
<td>GND (Ground, output): supplied by FH-002 hybrid</td>
</tr>
</tbody>
</table>

### Table 6: Pinout of the front panel serial I/O ports config. for RS-422

<table>
<thead>
<tr>
<th>Pin</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>TXD– (Transmit Data, output)</td>
</tr>
<tr>
<td>2</td>
<td>RTS– (Request to Send, output)</td>
</tr>
<tr>
<td>3</td>
<td>CTS+ (Clear to Send, input)</td>
</tr>
<tr>
<td>4</td>
<td>RXD+ (Receive Data, input)</td>
</tr>
<tr>
<td>5</td>
<td>GND (Signal GND)</td>
</tr>
<tr>
<td>6</td>
<td>TXD+ (Transmit Data, output)</td>
</tr>
<tr>
<td>7</td>
<td>RTS+ (Request to Send, output)</td>
</tr>
<tr>
<td>8</td>
<td>CTS– (Clear to Send, input)</td>
</tr>
<tr>
<td>9</td>
<td>RXD– (Receive Data, input)</td>
</tr>
</tbody>
</table>
IMPORTANT

In case of the RS-485 configuration connect the pins 3 and 7 to GND via the RS-485 cable, e.g. by connecting them to the pins 2 and 8, respectively.

Default port setup

- FH-002 installed for RS-232 support
- Asynchronous communication
- 9600 Baud, 8 data bits, 1 stop bit, no parity
- Hardware handshake protocol

Interface options

To easily vary the serial I/O interfaces according to the application’s needs FORCE COMPUTERS has developed RS-232, RS-422, and RS-485 hybrid modules: the FH-002, FH-003/FH-422T, and FH-007. The difference between FH-003 and FH-422T is that FH-422T has internal termination resistors. For each serial I/O port one of these 21-pin single in-line (SIL) hybrids is installed on-board:

- serial I/O port 1: hybrid installed in location J21
- serial I/O port 2: hybrid installed in location J22

Table 7 Pinout of the front panel serial I/O ports config. for RS-485

<table>
<thead>
<tr>
<th>Pin</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>RX-, TX-</td>
</tr>
<tr>
<td>2</td>
<td>GND</td>
</tr>
<tr>
<td>3</td>
<td>To be connected to GND via RS-485 cable</td>
</tr>
<tr>
<td>4</td>
<td>n.c.</td>
</tr>
<tr>
<td>5</td>
<td>GND</td>
</tr>
<tr>
<td>6</td>
<td>RX+, TX+</td>
</tr>
<tr>
<td>7</td>
<td>To be connected to GND via RS-485 cable</td>
</tr>
<tr>
<td>8</td>
<td>GND</td>
</tr>
<tr>
<td>9</td>
<td>n.c.</td>
</tr>
</tbody>
</table>
After installing the correct hybrid for the port under consideration the port has to be configured accordingly by using the appropriate switch setting. Thereby, the following options are selectable:

- FH-002 installed:
  - RS-232 asynchronous
  - RS-232 synchronous master
  - RS-232 synchronous slave
- FH-003/FH-422T installed:
  - RS-422
- FH-007 installed:
  - RS-485

The following switches apply to the port configuration:

- port 1: SW10-2, SW12-1, SW12-4 (see “SW10-2” on page 13),
- port 2: SW10-1, SW12-2, SW12-3 (see “SW10-1” on page 13).

## 2.8 SCSI

<table>
<thead>
<tr>
<th>Device: SCSI 53C720SE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency</td>
</tr>
<tr>
<td>Package</td>
</tr>
<tr>
<td>Accessible from</td>
</tr>
<tr>
<td>Access address</td>
</tr>
<tr>
<td>Port width</td>
</tr>
<tr>
<td>Interrupt request level</td>
</tr>
<tr>
<td>FGA-002 interrupt</td>
</tr>
</tbody>
</table>

The SCSI 53C720SE provides an 8-bit SCSI interface which is routed to the VMEbus P2 connector. The 8-bit SCSI interface at the VMEbus P2 is pinout compatible to the CPU-30 and CPU-40 (with EAGLE-01 or EAGLE-10/11).

The local bus interface is 32-bit wide and able to transfer data via the DMA controller of the SCSI 53C720SE.
The active termination can be selected by means of switches (see “SW11-2” and “SW11-1” on page 14). TERMPWR is supported.

Factory option

A 16-bit single-ended SCSI interface (wide SCSI) which is routed to the VMEbus P2 connector is available as factory option (see section 2.11 “VMEbus P2 Connector Pinout” on page 24). The wide-SCSI and the SERIAL-2-on-P2 factory option are not available simultaneously.

**SCSI Bus Termination**

**IMPORTANT**

According to the SCSI specification, the interconnecting flat cable must be terminated at both ends.

- Before connecting SCSI devices ensure correct SCSI bus termination:
  - If the CPU board is not located at either end of the cable, the termination must be disabled.
  - If the CPU board is located at the cable’s end, the termination must be enabled.

On the SYS68K/CPU-60 the termination of the SCSI bus is done by active terminators with a disconnect feature. This allows the outputs to be shut down to remove the terminator from the SCSI bus. It also reduces the standby power.

The disconnect input of the terminators is controlled by SW11-1 and SW11-2: default “OFF OFF = wide and 8-bit SCSI”, see page 14.

**SCSI bus terminator power**

The power for the terminator of any SCSI device will be provided from the CPU board directly, or from the SCSI bus itself. If the termination power is not delivered from any other SCSI device, it is delivered from the CPU board.

The TERMPWR (terminator power) supply from the CPU board is protected by a self-resetting fuse (1A max.) and a diode in series, as defined in the SCSI specification.

The on-board terminators draw power from the SCSI bus TERMPWR.
2.9 Floppy Disk – FDC

<table>
<thead>
<tr>
<th>Device: FDC 37C65C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency</td>
</tr>
<tr>
<td>Package</td>
</tr>
<tr>
<td>Accessible from</td>
</tr>
<tr>
<td>Access address</td>
</tr>
<tr>
<td>Port width</td>
</tr>
<tr>
<td>Interrupt request level</td>
</tr>
<tr>
<td>FGA-002 interrupt</td>
</tr>
</tbody>
</table>

The FDC signals are available at the VMEbus P2 connector (see section 2.11 “VMEbus P2 Connector Pinout” on page 24). An I/O back panel can be plugged onto the rear side of the backplane to interface to standard FDC connectors (see section 2.12 “SYS68K/IOBP-1” on page 28).

2.10 Ethernet – LAN

<table>
<thead>
<tr>
<th>Device: LAN AM 79C965A</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency</td>
</tr>
<tr>
<td>Package</td>
</tr>
<tr>
<td>Accessible from</td>
</tr>
<tr>
<td>Access address</td>
</tr>
<tr>
<td>Port width</td>
</tr>
<tr>
<td>Interrupt request level</td>
</tr>
<tr>
<td>FGA-002 interrupt</td>
</tr>
</tbody>
</table>

The Ethernet AUI interface is available at the front panel via a 15-pin D-Sub connector. As factory option Cheapernet is available via an SMB connector instead of the Ethernet AUI interface.
The CPU bus interface is 32-bit wide and able to transfer data via the DMA controller of the AM 79C965A.

The following table shows the pinout of the factory default Ethernet connector:

Table 8 15-pin AUI-Ethernet connector

<table>
<thead>
<tr>
<th>Pin</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>GND</td>
</tr>
<tr>
<td>2</td>
<td>Collision +</td>
</tr>
<tr>
<td>3</td>
<td>Transmit data +</td>
</tr>
<tr>
<td>4</td>
<td>GND</td>
</tr>
<tr>
<td>5</td>
<td>Receive data +</td>
</tr>
<tr>
<td>6</td>
<td>GND</td>
</tr>
<tr>
<td>7</td>
<td>n.c.</td>
</tr>
<tr>
<td>8</td>
<td>GND</td>
</tr>
<tr>
<td>9</td>
<td>Collision –</td>
</tr>
<tr>
<td>10</td>
<td>Transmit data –</td>
</tr>
<tr>
<td>11</td>
<td>GND</td>
</tr>
<tr>
<td>12</td>
<td>Receive data –</td>
</tr>
<tr>
<td>13</td>
<td>+12 V DC</td>
</tr>
<tr>
<td>14</td>
<td>GND</td>
</tr>
<tr>
<td>15</td>
<td>n.c.</td>
</tr>
</tbody>
</table>

Ethernet address The CPU board’s Ethernet address is displayed in the banner when entering FGA Boot.

2.11 VMEbus P2 Connector Pinout

I/O signals The I/O signal assignment on the VMEbus P2 connector allows interconnections using

- the SYS68K/IOBP-1 (8-bit SCSI, floppy disk, and serial I/O – see section 2.12 “SYS68K/IOBP-1” on page 28)
- and the IOPI-2 (8-bit SCSI, floppy disk, and serial I/O – see the IOPI-2 User’s Installation Manual).
IMPORTANT

In the following 2 figures unbracketed signals are available as factory default. Additionally,

- “*” marks the signals which are available with the wide SCSI factory option. They are implemented via 0-Ohm resistors.
- “**” marks the signals which are available with the FDC eject factory option.
- “***” marks the signals which are available with the SERIAL-2-on-P2 factory option.
- FDC DSEL1 is also available at C7 to provide backward compatibility to FDC DESL3.
- FDC DSEL2 is also available at C3 to provide backward compatibility to FDC DESL4.
- Instead of FDC DCHG, there formerly was FDC READY. However, the manufacturers of floppy disk drives have agreed upon not supporting the FDC READY signal any longer and using FDC DCHG (disk change) only.

Figure 3  P2 connector pinout with serial I/O config. for RS-232

<table>
<thead>
<tr>
<th>A</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCSI Data 0</td>
<td>1</td>
</tr>
<tr>
<td>SCSI Data 1</td>
<td>FDC RPM</td>
</tr>
<tr>
<td>SCSI Data 2</td>
<td>FDC HLOAD (FDC EJECT***)</td>
</tr>
<tr>
<td>SCSI Data 3</td>
<td>FDC DSEL2</td>
</tr>
<tr>
<td>SCSI Data 4</td>
<td>FDC DSEL1</td>
</tr>
<tr>
<td>SCSI Data 5</td>
<td>FDC DSEL2</td>
</tr>
<tr>
<td>SCSI Data 6</td>
<td>FDC DSEL1</td>
</tr>
<tr>
<td>SCSI Data 7</td>
<td>FDC MOTOR</td>
</tr>
<tr>
<td>SCSI DPA</td>
<td>FDC DIREC</td>
</tr>
<tr>
<td>GND</td>
<td>10</td>
</tr>
<tr>
<td>GND</td>
<td>FDC STEPX</td>
</tr>
<tr>
<td>GND</td>
<td>FDC WDATA</td>
</tr>
<tr>
<td>GND</td>
<td>FDC WGATE</td>
</tr>
<tr>
<td>TERMPWR</td>
<td>FDC TRKO0</td>
</tr>
<tr>
<td>GND</td>
<td>FDC WPROT</td>
</tr>
<tr>
<td>GND</td>
<td>15</td>
</tr>
<tr>
<td>GND</td>
<td>FDC RDATA</td>
</tr>
<tr>
<td>SCSI ATN</td>
<td>FDC SDSEL</td>
</tr>
<tr>
<td>GND</td>
<td>FDC DCHG</td>
</tr>
<tr>
<td>SCSI BSY</td>
<td>n.c.</td>
</tr>
<tr>
<td>SCSI ACK</td>
<td>n.c.</td>
</tr>
<tr>
<td>SCSI RST</td>
<td>20</td>
</tr>
<tr>
<td>n.c. (SCSI Data 8)</td>
<td></td>
</tr>
<tr>
<td>SCSI MSG</td>
<td>n.c. (SCSI Data 9)</td>
</tr>
<tr>
<td>SCSI SEL</td>
<td>n.c. (SCSI Data 10*)</td>
</tr>
<tr>
<td>SCSI CD</td>
<td>n.c. (SCSI Data 11*)</td>
</tr>
<tr>
<td>SCSI REQ</td>
<td>Serial DTR_2*** (SCSI Data 12*)</td>
</tr>
<tr>
<td>SCSI I/O</td>
<td>Serial DSR_2*** (SCSI Data 13*)</td>
</tr>
<tr>
<td>n.c. (Serial TxD_2***+*)</td>
<td>Serial RTS_2*** (SCSI Data 14*)</td>
</tr>
<tr>
<td>n.c. (Serial RXD_2***+*)</td>
<td>Serial CTS_2*** (SCSI Data 15*)</td>
</tr>
<tr>
<td>n.c. (Serial TXD_2***+*)</td>
<td>Serial DCD_2*** (SCSI DXP*)</td>
</tr>
<tr>
<td>Serial DSR_1</td>
<td>Serial DCD_1</td>
</tr>
<tr>
<td>Serial RTS_1</td>
<td>Serial RXD_1</td>
</tr>
<tr>
<td>Serial CTS_1</td>
<td>Serial TD_1</td>
</tr>
<tr>
<td>Serial GND_1</td>
<td>Serial DTR_1</td>
</tr>
</tbody>
</table>
**Figure 4**  P2 connector pinout with serial I/O config. for RS-422

<table>
<thead>
<tr>
<th>A</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCSI Data 0</td>
<td>1</td>
</tr>
<tr>
<td>SCSI Data 1</td>
<td>2</td>
</tr>
<tr>
<td>SCSI Data 2</td>
<td>3</td>
</tr>
<tr>
<td>SCSI Data 3</td>
<td>4</td>
</tr>
<tr>
<td>SCSI Data 4</td>
<td>5</td>
</tr>
<tr>
<td>SCSI Data 5</td>
<td>6</td>
</tr>
<tr>
<td>SCSI Data 6</td>
<td>7</td>
</tr>
<tr>
<td>SCSI Data 7</td>
<td>8</td>
</tr>
<tr>
<td>SCSI DPA</td>
<td>9</td>
</tr>
<tr>
<td>GND</td>
<td>10</td>
</tr>
<tr>
<td>GND</td>
<td>11</td>
</tr>
<tr>
<td>GND</td>
<td>12</td>
</tr>
<tr>
<td>TERMPWR</td>
<td>13</td>
</tr>
<tr>
<td>GND</td>
<td>14</td>
</tr>
<tr>
<td>GND</td>
<td>15</td>
</tr>
<tr>
<td>SCSI ATN</td>
<td>16</td>
</tr>
<tr>
<td>GND</td>
<td>17</td>
</tr>
<tr>
<td>SCSI BSY</td>
<td>18</td>
</tr>
<tr>
<td>SCSI ACK</td>
<td>19</td>
</tr>
<tr>
<td>SCSI RST</td>
<td>20</td>
</tr>
<tr>
<td>SCSI MSG</td>
<td>21</td>
</tr>
<tr>
<td>SCSI SEL</td>
<td>22</td>
</tr>
<tr>
<td>SCSI CD</td>
<td>23</td>
</tr>
<tr>
<td>SCSI REQ</td>
<td>24</td>
</tr>
<tr>
<td>SCSI I0</td>
<td>25</td>
</tr>
<tr>
<td>n.c. (Serial CTS+ 2***</td>
<td>26</td>
</tr>
<tr>
<td>n.c. (Serial RXD– 2***</td>
<td>27</td>
</tr>
<tr>
<td>n.c. (Serial RTS– 2***</td>
<td>28</td>
</tr>
<tr>
<td>Serial TXD– 1</td>
<td>29</td>
</tr>
<tr>
<td>Serial RTS– 1</td>
<td>30</td>
</tr>
<tr>
<td>Serial CTS– 1</td>
<td>31</td>
</tr>
<tr>
<td>Serial RXD– 1</td>
<td>32</td>
</tr>
</tbody>
</table>
### IMPORTANT

Serial 1, 2:

Note that

- the pins A30 and C31 must be connected to GND externally in case of the serial-1 RS-485 configuration.
- the pins A26 and C26 must be connected to GND externally in case of the serial-2 RS-485 configuration.

---

**Figure 5**  
P2 connector pinout with serial I/O config. for RS-485

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SCSI Data 0</strong></td>
<td>1</td>
<td><strong>FDC RPM</strong></td>
</tr>
<tr>
<td><strong>SCSI Data 1</strong></td>
<td>2</td>
<td><strong>FDC HLOAD (FDC EJECT</strong>)</td>
</tr>
<tr>
<td><strong>SCSI Data 2</strong></td>
<td>3</td>
<td><strong>FDC DSEL2</strong></td>
</tr>
<tr>
<td><strong>SCSI Data 3</strong></td>
<td>4</td>
<td><strong>FDC INDEX</strong></td>
</tr>
<tr>
<td><strong>SCSI Data 4</strong></td>
<td>5</td>
<td><strong>FDC DSEL1</strong></td>
</tr>
<tr>
<td><strong>SCSI Data 5</strong></td>
<td>6</td>
<td><strong>FDC DSEL2</strong></td>
</tr>
<tr>
<td><strong>SCSI Data 6</strong></td>
<td>7</td>
<td><strong>FDC DSEL1</strong></td>
</tr>
<tr>
<td><strong>SCSI Data 7</strong></td>
<td>8</td>
<td><strong>FDC MOTOR</strong></td>
</tr>
<tr>
<td><strong>SCSI DPA</strong></td>
<td>9</td>
<td><strong>FDC DIREC</strong></td>
</tr>
<tr>
<td><strong>GND</strong></td>
<td>10</td>
<td><strong>FDC STEPX</strong></td>
</tr>
<tr>
<td><strong>GND</strong></td>
<td>11</td>
<td><strong>FDC WDATA</strong></td>
</tr>
<tr>
<td><strong>GND</strong></td>
<td>12</td>
<td><strong>FDC WGATE</strong></td>
</tr>
<tr>
<td><strong>TERMWR</strong></td>
<td>13</td>
<td><strong>FDC TRK00</strong></td>
</tr>
<tr>
<td><strong>GND</strong></td>
<td>14</td>
<td><strong>FDC WPR0T</strong></td>
</tr>
<tr>
<td><strong>GND</strong></td>
<td>15</td>
<td><strong>FDC RDATA</strong></td>
</tr>
<tr>
<td><strong>SCSI ATN</strong></td>
<td>16</td>
<td><strong>FDC SDSEL</strong></td>
</tr>
<tr>
<td><strong>GND</strong></td>
<td>17</td>
<td><strong>FDC DCHG</strong></td>
</tr>
<tr>
<td><strong>SCSI BSY</strong></td>
<td>18</td>
<td>n.c.</td>
</tr>
<tr>
<td><strong>SCSI ACK</strong></td>
<td>19</td>
<td>n.c.</td>
</tr>
<tr>
<td><strong>SCSI RST</strong></td>
<td>20</td>
<td>n.c. (SCSI Data 8*)</td>
</tr>
<tr>
<td><strong>SCSI MSG</strong></td>
<td>21</td>
<td><strong>SCSI Data 9</strong></td>
</tr>
<tr>
<td><strong>SCSI SEL</strong></td>
<td>22</td>
<td>n.c. (SCSI Data 10*)</td>
</tr>
<tr>
<td><strong>SCSI CD</strong></td>
<td>23</td>
<td>n.c. (SCSI Data 11*)</td>
</tr>
<tr>
<td><strong>SCSI REQ</strong></td>
<td>24</td>
<td><strong>Serial RX+2, TX+2</strong></td>
</tr>
<tr>
<td><strong>SCSI IO</strong></td>
<td>25</td>
<td>n.c.</td>
</tr>
<tr>
<td>n.c.</td>
<td>26</td>
<td>n.c.</td>
</tr>
<tr>
<td>Serial RX-, TX-</td>
<td>27</td>
<td><strong>GND</strong></td>
</tr>
<tr>
<td><strong>GND</strong></td>
<td>28</td>
<td><strong>Serial RX-, TX-</strong></td>
</tr>
<tr>
<td>n.c.</td>
<td>29</td>
<td><strong>Serial RX-, TX-</strong></td>
</tr>
<tr>
<td>n.c.</td>
<td>30</td>
<td><strong>Serial RX-, TX-</strong></td>
</tr>
<tr>
<td><strong>GND</strong></td>
<td>31</td>
<td>n.c.</td>
</tr>
<tr>
<td>Serial RX+, TX+</td>
<td>32</td>
<td><strong>Serial RX+, TX+</strong></td>
</tr>
</tbody>
</table>
2.12 SYS68K/IOBP-1

As a separate price list item FORCE COMPUTERS offers a SYS68K/IOBP-1 I/O panel which is plugged into the VMEbus backplane from its rear.

**NOTICE**

To avoid damage to the board, do not use the SYS68K/IOBP-1 for the SYS68K/CPU-60 if serial port #2 is configured as RS-422.

The SYS68K/IOBP-1 enables easy connection to the I/O signals which are available on the CPU board’s P2 connector.

**Figure 6** SYS68K/IOBP-1 pin assignment for VME P2

<table>
<thead>
<tr>
<th>A</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>FDC Drive Select 4 (2)</td>
</tr>
<tr>
<td>2</td>
<td>FDC Drive Select 3 (1)</td>
</tr>
<tr>
<td>3</td>
<td>FDC Drive Select 2</td>
</tr>
<tr>
<td>4</td>
<td>FDC Motor On</td>
</tr>
<tr>
<td>5</td>
<td>FDC Direction In</td>
</tr>
<tr>
<td>6</td>
<td>FDC Step</td>
</tr>
<tr>
<td>7</td>
<td>FDC Write Data</td>
</tr>
<tr>
<td>8</td>
<td>FDC Write Gate</td>
</tr>
<tr>
<td>9</td>
<td>FDC Track 000</td>
</tr>
<tr>
<td>10</td>
<td>FDC Write Protect</td>
</tr>
<tr>
<td>11</td>
<td>FDC Read Data</td>
</tr>
<tr>
<td>12</td>
<td>FDC Side Select</td>
</tr>
<tr>
<td>13</td>
<td>FDC Disk Change *</td>
</tr>
<tr>
<td>14</td>
<td>FDC Disk Change</td>
</tr>
<tr>
<td>15</td>
<td>FDC Disk Change</td>
</tr>
<tr>
<td>16</td>
<td>FDC Disk Change</td>
</tr>
<tr>
<td>17</td>
<td>FDC Disk Change</td>
</tr>
<tr>
<td>18</td>
<td>FDC Disk Change</td>
</tr>
<tr>
<td>19</td>
<td>FDC Disk Change</td>
</tr>
<tr>
<td>20</td>
<td>GND</td>
</tr>
<tr>
<td>21</td>
<td>GND</td>
</tr>
<tr>
<td>22</td>
<td>GND</td>
</tr>
<tr>
<td>23</td>
<td>GND</td>
</tr>
<tr>
<td>24</td>
<td>GND</td>
</tr>
<tr>
<td>25</td>
<td>reserved</td>
</tr>
<tr>
<td>26</td>
<td>reserved</td>
</tr>
<tr>
<td>27</td>
<td>reserved</td>
</tr>
<tr>
<td>28</td>
<td>reserved</td>
</tr>
<tr>
<td>29</td>
<td>SER DSR</td>
</tr>
<tr>
<td>30</td>
<td>SER RTS</td>
</tr>
<tr>
<td>31</td>
<td>SER CTS</td>
</tr>
<tr>
<td>32</td>
<td>SER GND</td>
</tr>
<tr>
<td>33</td>
<td>SER DCD</td>
</tr>
<tr>
<td>34</td>
<td>SER RXD</td>
</tr>
<tr>
<td>35</td>
<td>SER TXD</td>
</tr>
<tr>
<td>36</td>
<td>SER DTR</td>
</tr>
</tbody>
</table>

* Instead of FDC DCHG, there formerly was FDC READY (see note on page 25).

The SYS68K/IOBP-1 contains the following connectors:

- P2 for the standard SCSI interface,
- P3 for the floppy disk interface,
- and P5 for the serial I/O port 1.

All row A and C pins of the VMEbus P2 connector are routed to the 64-pin male P4 connector on SYS68K/IOBP-1. However, the P4 connec-
tor pinout differs from the VME P2 connector by the counting direction: pin 1 of P4 = pin 32 of P2, ..., pin 32 of P4 = pin 1 of P2.
2.13 Testing the CPU Board Using VMEPROM

VMEPROM is a firmware providing a real-time multitasking multiuser monitor program. It is stored in the on-board system PROM.

Booting up VMEPROM To start VMEPROM automatically during power up or reset, the MODE 1 and MODE 2 rotary switches must both be set to F16. During booting FGA Boot is executed. After the successful pass of the self-test routine, the front-panel 7-segment hexadecimal display is switched off and its decimal point is periodically switched on and off.

POST codes If the SYS68K/CPU-60 fails during booting, the following POST (Power On SelfTest) codes indicate the status at the time of failure. The POST codes are displayed as status information during boot on the front-panel 7-segment hexadecimal display. The following table lists the POST codes in the order they occur during booting.

Table 9 POST codes indicating boot status

<table>
<thead>
<tr>
<th>POST code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cryptic code</td>
<td>When a ‘cryptic’ code is displayed, a general hardware error occurred. FGA Boot cannot be started.</td>
</tr>
<tr>
<td>&lt;Off&gt;</td>
<td>Read board ID from port and initialize 7-segment hexadecimal display.</td>
</tr>
<tr>
<td>0</td>
<td>Initialize the 68060 CPU registers CACR, ITTx, and DTTx, disable caches. FGA Boot has already left the boot-mode in this state.</td>
</tr>
<tr>
<td>1</td>
<td>Initialize the front-panel serial I/O port 1.</td>
</tr>
<tr>
<td>2</td>
<td>Initialize the CIO devices.</td>
</tr>
<tr>
<td>3</td>
<td>Identify board features and pre-select initialization sequence to follow. Read serial ID-ROM.</td>
</tr>
<tr>
<td>4</td>
<td>Determine CPU clock frequency (with cache enabled).</td>
</tr>
<tr>
<td>5</td>
<td>Determine capacity of main memory.</td>
</tr>
<tr>
<td>6</td>
<td>Verify local SRAM contents and store default values if checksum is wrong.</td>
</tr>
<tr>
<td>7</td>
<td>Perform auto-configuration (check hardware for special conditions such as being plugged in slot-1). If need, update SRAM value.</td>
</tr>
</tbody>
</table>
System controller

If the board is configured as system controller (i.e. SYS68K/CPU-60 is installed in slot 1), FGA Boot automatically enables the FGA-002 arbiter and switches on the user LED.

Starting a test after booting

To test the CPU board for correct operation enter the following command after the ? prompt:

? SELFTEST

SELFTEST does not provide a full-featured power-on self-test. However, it tests some I/O devices, the main memory, and the system timer tick interrupt. The time SELFTEST takes for testing depends on the main memory’s size. Allow approximately one minute per Mbyte.

Correct operation

After all tests have been done, the following message is displayed:

VMEPROM Hardware Selftest
-------------------------------
I/O test ........ passed
Memory test ...... passed
Clock test ...... passed

Table 9  POST codes indicating boot status (cont.)

<table>
<thead>
<tr>
<th>POST code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>Test for EAGLE modules (not applicable for SYS68K/CPU-60).</td>
</tr>
<tr>
<td>9</td>
<td>Read front-panel rotary switches and store to SRAM.</td>
</tr>
<tr>
<td>A</td>
<td>Check for firmware to start (default VMEPROM).</td>
</tr>
<tr>
<td>b</td>
<td>If the abort key is asserted or if there is no firmware to start, display the FGA Boot banner and start the shell.</td>
</tr>
<tr>
<td>C</td>
<td>Initialize FGA-002, arbiter, user LED and other hardware. Set up VMEbus A32 slave window (and A24 if enabled).</td>
</tr>
<tr>
<td>d</td>
<td>Clear DRAM (fill with 0) to initialize parity.</td>
</tr>
<tr>
<td>E</td>
<td>Call user program.</td>
</tr>
<tr>
<td>F</td>
<td>Try to execute the firmware.</td>
</tr>
<tr>
<td>&lt;Off&gt;</td>
<td>Left FGA Boot, started firmware.</td>
</tr>
</tbody>
</table>
3 Hardware

The SYS68K/CPU-60 is a high performance single-board computer providing a 32-bit master/slave VMEbus interface including DMA. It is based on

- the 68060 CPU (see section 3.4 “68060 CPU” on page 45),
- the FORCE gate array FGA-002 (see section 3.7 “FGA-002 Gate Array” on page 51),
- and the VMEbus (see section 3.15 “VMEbus Interface” on page 75).

Described features

The SYS68K/CPU-60 provides

- on-board shared DRAM (see section 3.8 “DRAM” on page 52)
- system PROM (see section 3.10 “System PROM” on page 62)
- boot PROM (see section 3.11 “Boot PROM” on page 64)
- on-board local SRAM (with on-board battery backup) (see section 3.13 “Local SRAM” on page 69) and optional on-board user SRAM (with on-board battery backup) (see section 3.9 “User SRAM (factory option)” on page 61)
- on-board real-time clock (with on-board battery backup) (see section 3.14 “Real-Time Clock – RTC 72423” on page 71)
- Ethernet interface, available at the front panel (see section 3.21 “Ethernet – LAN AM 79C965A” on page 96)
- single-ended SCSI interface and optional wide-fast-SCSI instead of the standard SCSI interface (see section 3.19 “SCSI – 53C720SE” on page 92)
- floppy interface (see section 3.20 “Floppy Disk – FDC 37C65C” on page 94)
- two RS-232 serial I/O ports (see section 3.18 “Serial I/O – SCC AM 85C30” on page 89)

DMA controllers

The following devices are collectively referred to as DMA controllers of the SYS68K/CPU-60 because they themselves provide an on-chip DMA controller:

- FGA-002 Gate Array,
- SCSI – 53C720SE,
- and Ethernet – LAN AM 79C965A.
### Front panel
The front panel of the SYS68K/CPU-60 provides an Ethernet port (see section 3.21 “Ethernet – LAN AM 79C965A” on page 96) and 2 serial ports (see section 3.18 “Serial I/O – SCC AM 85C30” on page 89). These ports serve as console port, for download and for data communication.

### Interfaces on VMEbus P2 connector
The following interfaces are available on the 3-row VMEbus P2 connector (see section 2.11 “VMEbus P2 Connector Pinout” on page 24):

- serial port 1 and 2 (see section 3.18 “Serial I/O – SCC AM 85C30” on page 89; note, however, that serial port 2 is only available if the wide SCSI factory option is not installed),
- the SCSI interface (see section 3.19 “SCSI – 53C720SE” on page 92),
- and the floppy interface (see section 3.20 “Floppy Disk – FDC 37C65C” on page 94).

### Factory options
The following factory options are available:

- capacity of DRAM (see section 3.8 “DRAM” on page 52 and “DRAM” on page 2)
- on-board user SRAM (with on-board battery backup) (see section 3.9 “User SRAM (factory option)” on page 61 and “User SRAM” on page 2)
- capacity of system PROM (see section 3.10 “System PROM” on page 62 and “System PROM” on page 2)
- capacity and type of boot PROM (see section 3.11 “Boot PROM” on page 64 and “Boot PROM” on page 2)
- capacity of local SRAM (see section 3.13 “Local SRAM” on page 69 and “Local SRAM” on page 3)
- capacity of user flash (see section 3.12 “User Flash” on page 68 and “User flash” on page 3)
- wide-fast-SCSI instead of the standard SCSI interface (see section 3.19 “SCSI – 53C720SE” on page 92): not available together with the SERIAL-2-on-P2 option.
- SERIAL-2-on-P2 option wide-fast-SCSI instead of the standard SCSI interface (see section 3.19 “SCSI – 53C720SE” on page 92): not available together with the SERIAL-2-on-P2 option.
Figure 7  SYS68K/CPU-60 block diagram

68060 CPU

记忆控制

CPU bus (060 bus)

RIALTO bus bridge (060/020)

FDC

RTC

Boot PROM

User flash (local flash)

FGA-002

I/O bus

AUX DMA

FDC

RTC

Local SRAM (NVRAM)

CIO

SCC

VL adaption

SCSI (+ SCSIbus Termin.)

4 LEDs
Hex. Displ.
2 rotary sw.
Reset key
Abort key

Ethernet LAN

VESA local bus (VL bus)

DRAM on-board

MEM-60 Memory bus

System PROM

User SRAM

VL adaption

SCSIbus

SCSIbus

MEM-60 Memory bus

68060 CPU

System PROM

User SRAM

VL adaption

SCSI (+ SCSIbus Termin.)

4 LEDs
Hex. Displ.
2 rotary sw.
Reset key
Abort key

Ethernet LAN

VESA local bus (VL bus)

DRAM on-board

MEM-60 Memory bus

System PROM

User SRAM

VL adaption

SCSI (+ SCSIbus Termin.)

4 LEDs
Hex. Displ.
2 rotary sw.
Reset key
Abort key

Ethernet LAN

VESA local bus (VL bus)
3.1 SYS68K/CPU-60 Memory Map

The SYS68K/CPU-60 is designed to utilize the entire 4-Gbyte address space of the 68060 CPU.
As the following table and section 3.15.3 “Master Interface: Data Transfer Size” on page 78 show the memory map of the SYS68K/CPU-60 is divided into address ranges for

- local memory,
- local I/O,
- FGA-002 internal registers,
- and the VMEbus address range (for the message broadcast area see table 43 “Address ranges related to AM codes” on page 77).

IMPORTANT

- Before erasing or programming the system PROM ensure that you do not destroy the VMEPROM image. The VMEPROM image resides in the first 512 Kbyte of the system PROM starting at address $FF00.0000_{16}$ and ending at $FF08.0000_{16}$.

- Before erasing or programming the boot PROM ensure that you do not destroy the FORCE COMPUTERS FGA Boot image. Before erasing or programming make a copy of the boot PROM device 1 in socket J70.

- The Ethernet – LAN AM 79C965A decodes itself and uses only the first 32 bytes. Accesses via this area are terminated by a bus error. All other register address spaces are mirrored.

- Always remember the following access rule for any reserved bits in any SYS68K/CPU-60 register: written as 0 read as undefined.

- All registers must be written or read using the data path width documented for the respective register.

- Always remember that in descriptions of data path widths byte refers to 8 bit, word to 16 bit, and long to 32 bit.
<table>
<thead>
<tr>
<th>Address range</th>
<th>Device</th>
<th>VMEbus accessible</th>
<th>Cache</th>
<th>Burst</th>
<th>Access width</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000.0000&lt;sub&gt;16&lt;/sub&gt;...003F.FFFF&lt;sub&gt;16&lt;/sub&gt;</td>
<td>DRAM: contributing to shared RAM, address range depends on memory capacity</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>32/16/8</td>
</tr>
<tr>
<td>00xx.0000&lt;sub&gt;16&lt;/sub&gt;...005F.FFFF&lt;sub&gt;16&lt;/sub&gt;</td>
<td>User SRAM (factory option): contributing to the shared RAM, consecutive to DRAM</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>32/16/8</td>
</tr>
<tr>
<td>00xx.0000&lt;sub&gt;16&lt;/sub&gt;...FAFF.FFFF&lt;sub&gt;16&lt;/sub&gt;</td>
<td>VME A32 extended address space (consecutive to DRAM and user SRAM)</td>
<td>n/a</td>
<td>N</td>
<td>Y</td>
<td>32/16/8</td>
</tr>
<tr>
<td>FB00.0000&lt;sub&gt;16&lt;/sub&gt;...FBFE.FFFF&lt;sub&gt;16&lt;/sub&gt;</td>
<td>VME A24 standard address space</td>
<td>n/a</td>
<td>N</td>
<td>Y</td>
<td>32/16/8</td>
</tr>
<tr>
<td>FF00.0000&lt;sub&gt;16&lt;/sub&gt;...FF3F.FFFF&lt;sub&gt;16&lt;/sub&gt;</td>
<td>System PROM: address range depends on system flash capacity</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>32/16/8</td>
</tr>
<tr>
<td>FF80.0C00&lt;sub&gt;16&lt;/sub&gt;...FF80.0DFF&lt;sub&gt;16&lt;/sub&gt;</td>
<td>SYS68K/CPU-60 Parameters and Timers – CIO Z8536:</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>8</td>
</tr>
<tr>
<td>FF80.0E00&lt;sub&gt;16&lt;/sub&gt;...FF80.0FFF&lt;sub&gt;16&lt;/sub&gt;</td>
<td>CIO1</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>8</td>
</tr>
<tr>
<td>FF80.1000&lt;sub&gt;16&lt;/sub&gt;</td>
<td>Slot-1 status register (RO)</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>8</td>
</tr>
<tr>
<td>FF80.2000&lt;sub&gt;16&lt;/sub&gt;...FF80.21FF&lt;sub&gt;16&lt;/sub&gt;</td>
<td>Serial I/O – SCC AM 85C30</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>8</td>
</tr>
<tr>
<td>FF80.3000&lt;sub&gt;16&lt;/sub&gt;...FF80.31FF&lt;sub&gt;16&lt;/sub&gt;</td>
<td>Real-Time Clock – RTC 72423</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>8</td>
</tr>
<tr>
<td>FF80.3800&lt;sub&gt;16&lt;/sub&gt;...FF80.39FF&lt;sub&gt;16&lt;/sub&gt;</td>
<td>Floppy Disk – FDC 37C65C</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>8</td>
</tr>
</tbody>
</table>
3.2 SYS68K/CPU-60 Interrupt Map

The FGA-002 monitors the VMEbus and all SYS68K/CPU-60 interrupt requests (IRQ):

- interrupt requests of all seven VMEbus interrupt levels,
- interrupt requests from on-board devices, e.g., from the SCSI and the floppy disk controller,
- and the FGA-002 specific interrupt requests.

ACFAIL* and SYSFAIL*

Additionally, the VMEbus signals ACFAIL* and SYSFAIL* can be programmed to interrupt the CPU on a software programmable level.

Flexible interrupt programming

Every interrupt source, including the VMEbus IRQs, can be programmed to interrupt the CPU on an individually programmable priority level, from 1 through 7. The FGA-002 may supply the interrupt vector, or it may initiate an interrupt vector fetch from the I/O device or from the VMEbus.
Interrupt vectors supplied by the FGA-002 all share a basic vector and a fixed vector offset for each source. The basic vector is software programmable.

The table below shows the local interrupt requests of the FGA-002 programmed for the local devices. For information on the vector offset and on programming the IRQ level, refer to the FORCE Gate Array FGA-002 User’s Manual.

### Table 11  
SYS68K/CPU-60 interrupt map

<table>
<thead>
<tr>
<th>Function</th>
<th>Device</th>
<th>FGA-002 IRQ</th>
<th>IRQ level</th>
<th>Vector supplied by</th>
</tr>
</thead>
<tbody>
<tr>
<td>Watchdog timer</td>
<td>Memory control</td>
<td>LIRQ0</td>
<td>sw. prog.</td>
<td>FGA-002</td>
</tr>
<tr>
<td>Floppy disk</td>
<td>FDC 37C65C</td>
<td>LIRQ1</td>
<td>sw. prog.</td>
<td>FGA-002</td>
</tr>
<tr>
<td>Timer 3</td>
<td>CIO Z8536 (CIO1)</td>
<td>LIRQ2</td>
<td>sw. prog.</td>
<td>FGA-002</td>
</tr>
<tr>
<td>Timer 2</td>
<td>CIO Z8536 (CIO1)</td>
<td>LIRQ3</td>
<td>sw. prog.</td>
<td>FGA-002</td>
</tr>
<tr>
<td>CPU board parameters</td>
<td>CIO Z8536 (CIO1 and CIO2)</td>
<td>LIRQ4</td>
<td>sw. prog.</td>
<td>CIO or FGA-002</td>
</tr>
<tr>
<td>SCC</td>
<td>SCC AM 85C30</td>
<td>LIRQ5</td>
<td>sw. prog.</td>
<td>SCC or FGA-002</td>
</tr>
<tr>
<td>SCSI</td>
<td>SCSI 53C720SE</td>
<td>LIRQ6</td>
<td>sw. prog.</td>
<td>FGA-002</td>
</tr>
<tr>
<td>Ethernet</td>
<td>LAN AM 79C965A</td>
<td>LIRQ7</td>
<td>sw. prog.</td>
<td>FGA-002</td>
</tr>
</tbody>
</table>
3.3 SYS68K/CPU-60 Parameters and Timers – CIO Z8536

The configuration and status information for several SYS68K/CPU-60 parameters and six 16-bit timers are accessible via 2 CIO Z8536 (see data sheet “CIO Z8536” in section 5).

**Parameters**

- **timers**
  - reading whether an interrupt request has been generated by one of the timers 2 or 3 from CIO1 (see table 13 “CIO1 port C data register” on page 41 and table 14 “CIO1 port B data register” on page 42),

- **DRAM**
  - reading the DRAM capacity (see table 13 “CIO1 port C data register” on page 41 and table 16 “CIO2 port C data register” on page 44),

- **ID-ROM**
  - controlling and reading the status of the serial ID-ROM signals (see table 14 “CIO1 port B data register” on page 42) for reading the CPU-board’s Ethernet address (see section 3.21 “Ethernet – LAN AM 79C965A” on page 96),

- **MODE x**
  - reading the setting of the 2 front-panel rotary switches (see table 15 “CIO1 port A data register (including MODE x status register)” on page 43),

- **A24-to-A32**
  - controlling the availability of the automatic A24 expansion (see table 16 “CIO2 port C data register” on page 44) and the A24-to-A32 address translation (see table 18 “CIO2 port A data register” on page 45),

- **board ID**
  - reading the CPU board identification number (see table 17 “CIO2 port B data register” on page 44),

- **DIAG**
  - controlling the front-panel DIAG 7-segment hexadecimal display (see table 17 “CIO2 port B data register” on page 44),

**CIO counters and timers**

CIO1 and CIO2 both offer 3 independently programmable 16-bit timers with 500 ns resolution which can also be used as counters. For information on CIO1 timer 2 and 3 see section 3.3.1 “MEM-60 DRAM Capacity and CIO1 Timer 3” on page 41 and section 3.3.2 “Flash VPP, Floppy Disk Control, and CIO1 Timer 2” on page 42.

**Clock**

The peripheral clock of both CIO devices is connected to a 4 MHz source.

**IRQ**

The interrupt request output of both CIO devices use LIRQ4 of the FGA-002. The interrupt vectors are supplied by the CIO devices. CIO1 has the higher interrupt priority in the daisy chain.

**CIO access**

The CIO devices are accessible via the 8-bit local I/O bus (byte mode).
### 3.3.1 MEM-60 DRAM Capacity and CIO1 Timer 3

#### Table 12

**SYS68K/CPU-60 parameters and timers register map and CIO loc.**

<table>
<thead>
<tr>
<th>Address</th>
<th>CIO device</th>
<th>On-board location</th>
<th>Register name</th>
</tr>
</thead>
<tbody>
<tr>
<td>FF80.0C0016</td>
<td>1</td>
<td>J35</td>
<td>CIO1 port C data reg.</td>
</tr>
<tr>
<td>FF80.0C0116</td>
<td></td>
<td></td>
<td>CIO1 port B data reg.</td>
</tr>
<tr>
<td>FF80.0C0216</td>
<td></td>
<td></td>
<td>CIO1 port A data reg.</td>
</tr>
<tr>
<td>FF80.0C0316</td>
<td></td>
<td></td>
<td>CIO1 ctrl. and pointer reg. (see data sheet “CIO Z8536” in section 5)</td>
</tr>
<tr>
<td>FF80.0E0016</td>
<td>2</td>
<td>J34</td>
<td>CIO2 port C data reg.</td>
</tr>
<tr>
<td>FF80.0E0116</td>
<td></td>
<td></td>
<td>CIO2 port B data reg.</td>
</tr>
<tr>
<td>FF80.0E0216</td>
<td></td>
<td></td>
<td>CIO2 port A data reg.</td>
</tr>
<tr>
<td>FF80.0E0316</td>
<td></td>
<td></td>
<td>CIO2 ctrl. and pointer reg. (see data sheet “CIO Z8536” in section 5)</td>
</tr>
</tbody>
</table>

#### Table 13

**CIO1 port C data register**

<table>
<thead>
<tr>
<th>FF80.0C0016</th>
<th>Bit</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>7</td>
<td>used as masking bits for write accesses to bit 3…0 (e.g.: if bit 4 is 1, bit 0 cannot be written)</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

**MC[2…0]** (RO) indicate the capacity of the DRAM installed on the MEM-60 memory module (see section 3.8.5 “Reading the DRAM Capacity” on page 57).

**T3IRQ** (W) **T3IRQ** controls the interrupt request output for timer 3 of CIO1. The 16-bit timer can generate interrupt requests at a software programmable level (the *FORCE Gate Array FGA-002 User’s Manual*). The corresponding interrupt request line is connected to the local IRQ #2 of the FGA-002. Additionally, the timer 3 of CIO1 can be programmed to generate an interrupt on the interrupt request line which is connected to the local IRQ #4 of the FGA-002. For information on the interpretation of the bit value of T3IRQ see data sheet “CIO Z8536” in section 5 and the *FORCE Gate Array FGA-002 User’s Manual*. 
3.3.2 Flash V_{PP}, Floppy Disk Control, and CIO1 Timer 2

<table>
<thead>
<tr>
<th>Table 14</th>
<th>CIO1 port B data register</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{FF80.0C01_{16}}</td>
<td>Bit 7</td>
</tr>
<tr>
<td>Value</td>
<td>ID_SCL</td>
</tr>
</tbody>
</table>

ID\_SCL (W) controls the ID-ROM SCL signal (I\textsuperscript{2}C bus), see “Ethernet node address” on page 97.

ID\_SDA (R/W) controls and indicates the status of the ID-ROM serial data signal (I\textsuperscript{2}C bus), see “Ethernet node address” on page 97.

FLVPP (W) controls whether the 12V programming voltage V\textsubscript{PP} for the flash memory (system PROM, boot PROM, and user flash) is turned on.

- = 0 V\textsubscript{PP} is turned on.
- = 1 V\textsubscript{PP} is turned off.

F\_xxxxxx (W) F\_DCHGEN, F\_ADDIR, F\_PCVAL, and F\_DRV control the FDC37C65C floppy disk drive interface (for the signals related to F\_DCHGEN, F\_PCVAL, and F\_DRV see section 3.20 “Floppy Disk – FDC 37C65C” on page 94).

- F\_DRV DRV signal
- F\_PCVAL PCVAL signal
- F\_ADDIR controls the AUX DMA transfer direction:
  - = 0 DMA write to FDC.
  - = 1 DMA read from FDC.
- F\_DCHGEN DCHGEN signal

T2IRQ (W) controls the interrupt request output for timer 2 of CIO1. The 16-bit timer can generate interrupt requests at a software programmable level. Timer 2 can be linked with timer 1 to establish a 32-bit timer. The corresponding interrupt request line is connected to the local IRQ #3 of the FGA-002. Additionally, the timer 2 of CIO1 can be programmed to generate an interrupt on the interrupt request line which is connected to the local IRQ #4 of the FGA-002. For information on the interpretation of the bit value of T2IRQ see data sheet “CIO Z8536” in section 5 and the \textit{FORCE Gate Array FGA-002 User’s Manual}. 
3.3.3 MODE x Rotary Switch Setting

Rotary switches
– reset and abort

The MODE x rotary switches serve a special function in conjunction with the reset and abort keys. This function is built into the boot PROM and is described in detail in the boot software description of the *FORCE Gate Array FGA-002 User’s Manual*.

– applications

For application programs, the rotary switches can be used as a general purpose input channel for diagnostics, configuration selection, or automatic system boot with different configurations.

– VMEPROM

VMEPROM uses the rotary switches for automatic configuration (see section 6.2.3 “Rotary Switches” on page 128).

### Table 15 CIO1 port A data register (including MODE x status register)

<table>
<thead>
<tr>
<th>Bit</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>RS2B[3…0]</td>
<td>RS1B[3…0]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**RS2B[3…0] (RO)**

RS2B[3…0] is commonly referred to as MODE 2 status register indicating the setting of the MODE 2 front-panel rotary switch:

- \( \text{F}_{16} \) MODE 2 is set to F.
- \( \text{E}_{16} \) MODE 2 is set to E.
- ... ...

**RS1B[3…0] (RO)**

RS1B[3…0] is commonly referred to as MODE 1 status register indicating the setting of the MODE 1 front-panel rotary switch:

- \( \text{F}_{16} \) MODE 1 is set to F.
- \( \text{E}_{16} \) MODE 1 is set to E.
- ... ...
3.3.4 On-board DRAM Capacity and Automatic A24 Expansion

Table 16  CIO2 port C data register

<table>
<thead>
<tr>
<th>Bit</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>used as masking bits for write accesses to bit 3…0 (e.g.: if bit 4 is 1, bit 0 cannot be written)</td>
<td>MC[2…0]</td>
<td>A24E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

MC[2…0] (RO) indicate the capacity of the installed on-board DRAM (see section 3.8.5 “Reading the DRAM Capacity” on page 57).

A24E (W)

= 0 A24 expansion enabled (A24 and A32 enabled).

= 1 A24 expansion disabled (only A32 enabled).

3.3.5 Board ID and DIAG Display

Table 17  CIO2 port B data register

<table>
<thead>
<tr>
<th>Bit</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>DP</td>
<td>SEG_G</td>
<td>SEG_F</td>
<td>SEG_E</td>
<td>SEG_D</td>
<td>SEG_C</td>
<td>SEG_B</td>
<td>SEG_A</td>
</tr>
</tbody>
</table>

DP and SEG_G...SEG_A (R/W)

When reading these bits directly after power-up or reset these bits indicate the CPU board identification number which is assigned to every type of CPU board. The CPU identification number does not identify the factory options which might be available for CPU speeds, memory capacity, or installed modules. In case of the SYS68K/CPU-60 the CPU board identification number is $40_{10} = 28_{16} = 10.1000_2$.

After the first reading of these bits the bits control the status of the decimal point (DP) and the segments (SEG_G...SEG_A) in the front-panel hexadecimal display (see figure below for naming conventions).

= 0 The respective part of the display is turned off.

= 1 The respective part of the display is turned on.
3.3.6 A24-to-A32 Address Translation

If the automatic A24 expansion is enabled (see table 16 “CIO2 port C data register” on page 44), A[31...24] control the status of the A31...24 address lines (see section 3.15.7 “Slave Interface: Address Modifier Decoding and A24 Slave Mode” on page 81).

3.4 68060 CPU

The 68060 CPU is one of the fundamental components of the SYS68K/CPU-60. Therefore, the M68060 User’s Manual is delivered together with this Technical Reference Manual.

3.4.1 Hardware Interface of the 68060 CPU

The 68060 CPU uses a non-multiplexed address and data bus. The bus interface supports synchronous data transfers between the CPU and other devices in the system.

The CPU drives the address signals (A0 – A31), the size signals (SIZ0, SIZ1) and the transfer cycle modifier (TM0 – TM2) on every cycle, independently of a cache hit or miss. These signals are used to decode the memory map of the CPU board.

The hardware on the CPU board is notified by the address and data strobe signals that the current cycle is not a cache cycle and that the decoding outputs are strobed to be valid.

The 32 data lines (D0 – D31) are also driven by the 68060 CPU on write cycles.
CPU sensed signals  The 32 data lines (D0 – D31) are sensed on read cycles. The size of the data transfer is defined by the SIZE output signals (always driven by the 68060 CPU). Cycles are acknowledged by the transfer acknowledge (TA) signal.

Bus error generation  In case of bus operation a bus error will be generated if a device does not respond correctly.

Bus error sensing  A bus error is sensed by the CPU via the TEA signal. If a bus error occurs, the current cycle is aborted (illegal transfer or incorrect data) and exception handling starts. VMEbus transfers may also be aborted via TEA.

If TA and TEA are sensed simultaneously, the CPU enters the retry bus operation sequence. A retry happens, whenever the CPU tries to access a device on the I/O bus or the VMEbus and an external master accesses the shared RAM simultaneously.

3.4.2 Instruction Set of the 68060 CPU

For the 68060 CPU instruction set and for further information concerning programming, refer to the 68060 User’s Manual.

3.4.3 Vector Table of the 68060 CPU

This table lists all vectors defined and used by the 68060 CPU.

Table 19  68060 CPU exception vector assignments

<table>
<thead>
<tr>
<th>Vector number(s)</th>
<th>Vector offset (Hex)</th>
<th>Assignment</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>000&lt;sub&gt;16&lt;/sub&gt;</td>
<td>Reset initial interrupt stack pointer</td>
</tr>
<tr>
<td></td>
<td>004&lt;sub&gt;16&lt;/sub&gt;</td>
<td>Reset initial program counter</td>
</tr>
<tr>
<td>1</td>
<td>008&lt;sub&gt;16&lt;/sub&gt;</td>
<td>Access fault (bus error)</td>
</tr>
<tr>
<td></td>
<td>00C&lt;sub&gt;16&lt;/sub&gt;</td>
<td>Address error</td>
</tr>
<tr>
<td>2</td>
<td>010&lt;sub&gt;16&lt;/sub&gt;</td>
<td>Illegal instruction</td>
</tr>
<tr>
<td>3</td>
<td>014&lt;sub&gt;16&lt;/sub&gt;</td>
<td>Integer divide by zero</td>
</tr>
<tr>
<td>4</td>
<td>018&lt;sub&gt;16&lt;/sub&gt;</td>
<td>CHK, CHK2 instruction</td>
</tr>
<tr>
<td>5</td>
<td>01C&lt;sub&gt;16&lt;/sub&gt;</td>
<td>TRAPcc, TRAPV instructions</td>
</tr>
<tr>
<td>6</td>
<td>020&lt;sub&gt;16&lt;/sub&gt;</td>
<td>Privilege violation</td>
</tr>
<tr>
<td>7</td>
<td>024&lt;sub&gt;16&lt;/sub&gt;</td>
<td>Trace</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 19  68060 CPU exception vector assignments (cont.)

<table>
<thead>
<tr>
<th>Vector number(s)</th>
<th>Vector offset (Hex)</th>
<th>Assignment</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>028₁₆</td>
<td>Line 1010 emulator (unimplemented A-line opcode)</td>
</tr>
<tr>
<td></td>
<td>02C₁₆</td>
<td>Line 1111 emulator (unimplemented F-line opcode)</td>
</tr>
<tr>
<td>12</td>
<td>030₁₆</td>
<td>Emulator interrupt</td>
</tr>
<tr>
<td>13</td>
<td>034₁₆</td>
<td>unused by MC68060</td>
</tr>
<tr>
<td>14</td>
<td>038₁₆</td>
<td>Format error</td>
</tr>
<tr>
<td>15</td>
<td>03C₁₆</td>
<td>Uninitialized interrupt</td>
</tr>
<tr>
<td>16 – 23</td>
<td>040₁₆...05C₁₆</td>
<td>reserved (unassigned)</td>
</tr>
<tr>
<td>24</td>
<td>060₁₆</td>
<td>Spurious interrupt</td>
</tr>
<tr>
<td></td>
<td>064₁₆</td>
<td>Interrupt autovector for level</td>
</tr>
<tr>
<td></td>
<td>068₁₆</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>06C₁₆</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>070₁₆</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>074₁₆</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>078₁₆</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>07C₁₆</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>080₁₆...0BC₁₆</td>
<td>TRAP #0 – 15 instruction vectors</td>
</tr>
<tr>
<td>48</td>
<td>0C0₁₆</td>
<td>FPCP</td>
</tr>
<tr>
<td>49</td>
<td>0C4₁₆</td>
<td>branch or set on unordered condition</td>
</tr>
<tr>
<td>50</td>
<td>0C8₁₆</td>
<td>inexact result</td>
</tr>
<tr>
<td>51</td>
<td>OCC₁₆</td>
<td>divide by zero</td>
</tr>
<tr>
<td>52</td>
<td>ODO₁₆</td>
<td>underflow</td>
</tr>
<tr>
<td>53</td>
<td>OD₄₁₆</td>
<td>operand error</td>
</tr>
<tr>
<td>54</td>
<td>ODB₁₆</td>
<td>overflow</td>
</tr>
<tr>
<td>55</td>
<td>ODC₁₆</td>
<td>signalling SNAN</td>
</tr>
<tr>
<td>56</td>
<td>0E0₁₆</td>
<td>unimplemented data type</td>
</tr>
<tr>
<td>57</td>
<td>0E4₁₆</td>
<td>unused by MC68060</td>
</tr>
<tr>
<td>58</td>
<td>0E8₁₆</td>
<td>Defined for 68852, unused by 68060</td>
</tr>
<tr>
<td>59</td>
<td>0EC₁₆</td>
<td>Defined for 68852, unused by 68060</td>
</tr>
<tr>
<td>60</td>
<td>0F0₁₆</td>
<td>reserved (unassigned)</td>
</tr>
<tr>
<td>61</td>
<td>0F₄₁₆</td>
<td>Unimplemented effective address</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Unimplemented integer instruction</td>
</tr>
<tr>
<td>62 – 63</td>
<td>0F8₁₆...0FC₁₆</td>
<td>reserved (unassigned)</td>
</tr>
<tr>
<td>64 – 255</td>
<td>100₁₆...3FC₁₆</td>
<td>User defined vectors (192)</td>
</tr>
</tbody>
</table>
3.5 Watchdog Timer

There is a watchdog timer installed on the SYS68K/CPU-60 to monitor the 68060 CPU activity. The watchdog timer is able

- to issue an interrupt to the 68060 CPU
- and to generate a pseudo power up pulse whereby the CPU board is reset but in contrast to a normal power up the on-board LCAs are not loaded from the serial PROM.

One timeout is specified for both actions.

**IRQ**
The watchdog interrupt is the LIRQ0 input of the FGA-002.

**IMPORTANT**
This input must be configured
- to be sensitive on a falling edge signal
- and to generate level-7 interrupts to the 68060 CPU.

3.5.1 Watchdog Operation

The SYS68K/CPU-60 watchdog timer monitors the 68060 CPU activity by awaiting a trigger event from the 68060 CPU within the watchdog timer’s timeout period.

**Trigger event**
The watchdog timer is triggered by setting the RESTART bit in the watchdog retrigger register to 0 (see table 21 “Watchdog retrigger register (WDR)” on page 49).

**Timeout**
The watchdog timeout is selectable by setting the WDTIME bit in the memory configuration register (see table 26 “MCR, memory configuration register” on page 54): either 40 ms (± 30 %) or 0.5 s (± 30 %).

**Starting the watchdog**
The watchdog timer is started by setting the ENWD bit in the memory configuration register (see table 26 “MCR, memory configuration register” on page 54). Once started, it cannot be stopped unless a reset occurs. In case of a reset the watchdog timer is automatically disabled.

If the retrigger event occurs within the watchdog timeout period, the watchdog timer is restarted.

**NMI generation**
If the retrigger event does not occur within the watchdog timeout period, the watchdog timer generates an NMI to the 68060 CPU.
If the retrigger event occurs within the watchdog timeout period after generating the NMI, the watchdog timer is restarted.
timeout period after NMI – reset

If the retrigger event does not occur within the watchdog timeout period after generating the NMI, the watchdog timer generates a pseudo power up pulse, thereby automatically stopping itself. It then must be restarted by setting the ENWD bit in the memory configuration register as already stated above.

Reset

To enable detection of a watchdog reset the watchdog NMI handler has to clear the WDIRQ bit in the memory diagnostic register on every watchdog interrupt (see table 27 “Memory diagnostic register (MDR)” on page 55). A watchdog reset can then be detected by reading the WDIRQ bit.

3.5.2 Watchdog Register Map

The watchdog timer is controlled by the contents of the following registers:

Table 20 Watchdog register map (superset of the memory controller register map)

<table>
<thead>
<tr>
<th>Address</th>
<th>Register name and access</th>
</tr>
</thead>
<tbody>
<tr>
<td>FFF4.000816</td>
<td>Memory configuration reg. (MCR), R/W, see table 26 “MCR, memory configuration register” on page 54</td>
</tr>
<tr>
<td>FFF4.000A16</td>
<td>Watchdog retrigger reg. (WDR), WO</td>
</tr>
<tr>
<td>FFF4.000B16</td>
<td>Memory diagnostic reg. (MDR), RO, see table 27 “Memory diagnostic register (MDR)” on page 55</td>
</tr>
</tbody>
</table>

Table 21 Watchdog retrigger register (WDR)

<table>
<thead>
<tr>
<th>Bit</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>reserved</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**RESTART**

**RESTART** retriggers the watchdog timer.

= 0 Retrigger the watchdog timer.

= 1 No action is taken.
3.6 RIALTO Bus Bridge

The bus bridge is intended to maximize the performance of the CPU board. As data and address bridge between the 68040-type CPU bus and the FGA-002 interface chip the RIALTO bus bridge is especially designed to support fast VMEbus master/slave block transfers.

Revision of RIALTO bus bridge

After power up the SNOOP [2..0] bits in the bridge configuration register (BCR) show the revision of the RIALTO bus bridge (see section 3.6.2 “Bridge Configuration Register” on page 50).

3.6.1 Register Set

The following register map shows all internal registers and their corresponding register addresses.

Table 22 RIALTO bus bridge register map

<table>
<thead>
<tr>
<th>Offset addr.</th>
<th>Reset value</th>
<th>Register name</th>
</tr>
</thead>
<tbody>
<tr>
<td>FFF4.000016</td>
<td>0000.000016</td>
<td>reserved</td>
</tr>
<tr>
<td>FFF4.000416</td>
<td>00xx.xxxx16</td>
<td>Bridge config. reg. (BCR)</td>
</tr>
</tbody>
</table>

3.6.2 Bridge Configuration Register

The bridge configuration register of the RIALTO bus bridge features several status and control bits to monitor and control the configuration.

Table 23 Bridge configuration register (BCR)

<table>
<thead>
<tr>
<th>FFF4.000416</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit</td>
</tr>
<tr>
<td>Value</td>
</tr>
</tbody>
</table>

LANDEC (R/W)

LANDEC defines the Ethernet decoding space (see section 3.21.1 “Register Access” on page 98).

= 0 Ethernet decoding space is FFF0.000016...FFFF.FFFF16.
= 1 Ethernet decoding space is 0000.000016...0003.FFFF16.
USERLED (R/W)

Controls the front-panel UL LED.

= 0  LED is on.
= 1  LED is off.

SNOOP[2..0] (R/W)

The snoop bits shrink the DRAM accessible from the VMEbus to the value listed in the table. This is done by masking out the higher address lines. After power up the SNOOP[2..0] bits show the revision of the RIALTO bus bridge.

Table 24  Snoop window definition in BCR

<table>
<thead>
<tr>
<th>SNOOP[2..0]</th>
<th>Snoop window</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0</td>
<td>32 Mbyte</td>
</tr>
<tr>
<td>0 0 1</td>
<td>16 Mbyte</td>
</tr>
<tr>
<td>0 1 0</td>
<td>8 Mbyte</td>
</tr>
<tr>
<td>0 1 1</td>
<td>4 Mbyte</td>
</tr>
<tr>
<td>1 0 0</td>
<td>2 Mbyte</td>
</tr>
<tr>
<td>1 0 1</td>
<td>256 Mbyte</td>
</tr>
<tr>
<td>1 1 0</td>
<td>128 Mbyte</td>
</tr>
<tr>
<td>1 1 1</td>
<td>64 Mbyte</td>
</tr>
</tbody>
</table>

3.7  FGA-002 Gate Array

The FGA-002 controls the I/O bus and builds the interface to the VMEbus. It also includes

- a DMA controller,
- complete interrupt management,
- a message broadcast interface (FMB),
- timer functions,
- and mailbox locations.

Monitoring the 020 bus

The FGA-002 monitors the 020 bus. When any local device is accessed the FGA-002 takes charge of all control signals in addition to the address and data signals used.

Managing the VMEbus

The FGA-002 serves as manager for the VMEbus. All VMEbus address and data lines are connected to the gate array via buffers. Additional functions such as the VMEbus interrupt handler and arbiter are also installed on the FGA-002.
The start address of the FGA-002 registers is \texttt{FFD0.0000}_{16}.

For a detailed description of the FGA-002 registers, see the \textit{FORCE Gate Array FGA-002 User’s Manual}.

### 3.8 DRAM

The CPU board provides shared dynamic RAM (DRAM). For the available capacity options see “DRAM” on page 2. The DRAM capacity currently installed is software readable (see section 3.8.5 “Reading the DRAM Capacity” on page 57).

The DRAM is optimized for fast accesses from the 68060 CPU and the SCSI and Ethernet DMA controllers (see section 3.8.4 “DRAM Performance” on page 56). Snooping is supported.

#### IMPORTANT

To guarantee the cache coherence of the DRAM, it is necessary to configure the snoop window in the bridge configuration register of the RIALTO bus bridge (see section 3.8.7 “Cache Coherence and Snooping” on page 58).

#### Accessibility

The DRAM is accessible from the

- 68060 CPU (including burst mode support),
- FGA-002 Gate Array DMA controller,
- SCSI – 53C720SE DMA controller (including burst mode support),
- Ethernet – LAN AM 79C965A DMA controller,
- and also from other VMEbus masters.

#### Burst mode

Burst mode support is always enabled. Advanced on-board memory control logic routes data to and from the 68060 CPU, the SCSI controller, and the VMEbus interface.

#### DRAM read, parity support

For every read cycle all 32 data and all 4 parity bits are read from the DRAM, regardless of size (byte, word, long-word, or cache line) and regardless of master (68060 CPU, DMA controllers, or VMEbus). The 32 data and 4 parity bits are stored in the memory controller. Parity is regenerated in the memory controller and compared to the parity bits read from memory. If a parity error is detected for an accessed byte, a bus error acknowledge is generated and a parity-error flag is set in the memory controller (see table 27 “Memory diagnostic register (MDR)” on page 55).
DRAM write

Write cycles are handled differently:
In case of a long-word access, the DRAM can be written immediately – including the parity info generated by the memory control. A transfer acknowledge (TA) signal is asserted and the cycle completed.
For all other write cycles (byte, word) the currently valid parity bits stored in the DRAM must be read at first. In order to satisfy this condition, only the necessary data will be written, the remaining data already stored in DRAM memory will stay unmodified. Additionally, the new parity bits generated by the memory controller will be merged with the parity bits read from DRAM memory and finally all four parity bits are written to DRAM memory. The transfer acknowledge (TA) signal is asserted and the cycle completed.

Write posting

All write cycles are terminated before they are fully processed so that the master which is writing to DRAM can continue its operations (write posting).

3.8.1 Register Set

The following register map shows all internal registers and their corresponding register addresses.

<table>
<thead>
<tr>
<th>Table 25</th>
<th>Memory controller register map (included in the watchdog register map)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Offset addr.</td>
<td>Reset value</td>
</tr>
<tr>
<td>FFF4.0000B_{16}</td>
<td>xx00.00xx.xx0.0000_{2}</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>FFF4.0000B_{16}</td>
<td>xxxxxx.xx00_{16}</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### 3.8.2 Memory Configuration Register

The memory configuration register provides several bits to control the configuration of the memory controller.

#### Table 26 MCR, memory configuration register

<table>
<thead>
<tr>
<th>FFF4.0000</th>
<th>15..14</th>
<th>13</th>
<th>12</th>
<th>11</th>
<th>10</th>
<th>9..5</th>
<th>4..2</th>
<th>1..0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit</td>
<td>Value</td>
<td>WDTIME</td>
<td>ENWD</td>
<td>RESETOUT</td>
<td>ENPAR</td>
<td>reserved</td>
<td>VERSMEMCTRL [4..2]</td>
<td>VMEBUS_TIMER [1..0]</td>
</tr>
<tr>
<td>WDTIME</td>
<td>reserved</td>
<td>WD TIME</td>
<td>ENWD</td>
<td>RESETOUT</td>
<td>ENPAR</td>
<td>reserved</td>
<td>VERSMEMCTRL [4..2]</td>
<td>VMEBUS_TIMER [1..0]</td>
</tr>
<tr>
<td>(R/W)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>= 0 (default)</td>
<td>40 ms (± 30 %)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>= 1</td>
<td>0.5 s (± 30 %)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ENWD</td>
<td>ENWD starts the watchdog timer if set to 1. This bit can only be set to 1 or read (setting it to 0 is impossible). A reset of the SYS68K/CPU-60 clears this bit automatically.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(R/W)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>= 0</td>
<td>Watchdog timer is disabled.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>= 1</td>
<td>Watchdog timer has been started.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RESETOUT</td>
<td>RESETOUT controls the generation of a reset. If the RESETOUT bit in the memory configuration register is set to 1, a reset is generated. Setting the RESETOUT bit has the same effect as a reset generated by the watchdog timer.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(R/W)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>= 0</td>
<td>No action is taken.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>= 1</td>
<td>Generates a reset.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ENPAR</td>
<td>ENPAR controls whether DRAM parity check is enabled (see “ENPARIN” on page 55).</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(R/W)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>= 0</td>
<td>DRAM parity check is disabled.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>= 1</td>
<td>DRAM parity check is enabled.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VERSMEMCTRL [4..2] (RO)</td>
<td>VERSMEMCTRL indicates the version memory control.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>= 0</td>
<td>8th revision of memory control</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>= ...</td>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>= 7</td>
<td>1st revision of memory control.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
VMEBUSTIMER [1..0] controls the VME bustimer timeout (see section 3.17.5 “VMEbus Timer” on page 89).

- **00** The VME bustimer is disabled.
- **01** timeout = 82 µs (±10 %) for 25 MHz CPU bus frequency
timeout = 65 µs (±10 %) for 33 MHz CPU bus frequency
- **10** timeout = 164 µs (±10 %) for 25 MHz CPU bus frequency
timeout = 130 µs (±10 %) for 33 MHz CPU bus frequency
- **11** timeout = 328 µs (±10 %) for 25 MHz CPU bus frequency
timeout = 260 µs (±10 %) for 33 MHz CPU bus frequency

### 3.8.3 Memory Diagnostic Register

The memory diagnostic register provides the monitoring of several status flags.

**IMPORTANT** A write access to MDR or a power up clears bit 3…0 in the MDR. They are not cleared on a normal reset.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>SEL25M</td>
</tr>
<tr>
<td>6</td>
<td>ENPARIN</td>
</tr>
<tr>
<td>5</td>
<td>SIBK</td>
</tr>
<tr>
<td>4</td>
<td>WESYSFLASH</td>
</tr>
<tr>
<td>3</td>
<td>WDIRQ</td>
</tr>
<tr>
<td>2</td>
<td>CPUBT</td>
</tr>
<tr>
<td>1</td>
<td>BTF</td>
</tr>
<tr>
<td>0</td>
<td>PERR</td>
</tr>
</tbody>
</table>

**SEL25M** SEL25M indicates the 68060 CPU speed.

- **0** 33 MHz.
- **1** 25 MHz.

**ENPARIN** ENPARIN indicates the setting of SW9-2 for DRAM parity check (see “ENPAR (R/W)” on page 54). When reading this bit the software should set the ENPAR bit accordingly. FGA Boot conforms to this rule and enables or disables DRAM parity check according to the setting of the ENPARIN bit.

- **0** DRAM parity check should be disabled by software.
- **1** DRAM parity check should be enabled by software.

**SIBK** SIBK indicates whether a single or both DRAM banks are assembled.

- **0** Both DRAM banks are assembled.
- **1** Only DRAM bank 1 is assembled.
### WESYSFLASH

WESYSFLASH indicates the current setting of SW10-3, thereby indicating whether write access to the system PROM is enabled.

- **= 0** SW10-3 is set to ON.
- **= 1** SW10-3 is set to OFF (OFF = writing enabled).

### WDIRQ

WDIRQ flags that a watchdog interrupt has been generated. To enable detection of a watchdog reset the watchdog NMI handler has to clear WDIRQ on every watchdog interrupt. A watchdog reset can then be detected by reading the WDIRQ bit.

- **= 0** No watchdog interrupt occurred.
- **= 1** A watchdog interrupt has been generated.

### CPUBT

CPUBT indicates whether a bus error occurred – the CPU bustimer terminates cycles on the CPU bus by generating a timeout bus error.

- **= 0** No bus error occurred.
- **= 1** A bus error has been generated.

### BTF

BTF, write burst to flash. This bit is set in case of a write burst to the system flash.

- **= 0** No BTF bus error detected.
- **= 1** A BTF bus error has occurred.

### PERR

PERR, parity bus error. This bit is set whenever a parity error is detected.

- **= 0** No parity error detected.
- **= 1** A parity error has occurred.

### 3.8.4 DRAM Performance

The on-board memory control logic is optimized for fast accesses from the 68060 CPU providing the maximum performance. Since the 68060 CPU contains an on-chip data and instruction cache many CPU accesses are cache line "burst fills". Within four 4-byte cycles these burst fills attempt to read 16 consecutive bytes into the 68060 CPU.

#### "5-1-1-1" burst transfer

The first read cycle of such a burst usually requires 5 CPU clock cycles (200 ns at 25 MHz). Due to the optimized design of the memory control logic, each subsequent cycle only requires 1 CPU clock cycle (40 ns) to complete. This is commonly called a "5-1-1-1" burst transfer. Overall, the total cache line "burst fill" operation requires 8 clock cycles to transfer 16 bytes, providing a memory bandwidth of over 50 Mbyte/s.

#### Single read and write

Not all CPU accesses are burst transfers. Single read and write transactions are also supported at maximum speed. A single read or write access (1, 2, or 4 bytes) requires 5 CPU clock cycles. Distributed asynchronous
refresh is provided every 14 μs and an access during a pending refresh cycle may be delayed by a maximum of 5 additional clock cycles.

### 3.8.5 Reading the DRAM Capacity

The installed on-board and the MEM-60 DRAM capacity are encoded in 3 bits (see MC[2…0] in “CIO2 port C data register” on page 44 and MC[2…0] in “CIO1 port C data register” on page 41, respectively).

**Table 28** DRAM capacity encoding at CIOx port C data registers

<table>
<thead>
<tr>
<th>MC[2…0]</th>
<th>DRAM capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• on-board (when reading MC[2…0] from CIO2 port C data register)</td>
</tr>
<tr>
<td></td>
<td>• on MEM-60 (when reading MC[2…0] from CIO1 port C data register)</td>
</tr>
<tr>
<td>0 0 0</td>
<td>32 Mbyte</td>
</tr>
<tr>
<td>0 0 1</td>
<td>16 Mbyte</td>
</tr>
<tr>
<td>0 1 0</td>
<td>8 Mbyte</td>
</tr>
<tr>
<td>0 1 1</td>
<td>4 Mbyte</td>
</tr>
<tr>
<td>1 0 0</td>
<td>0 Mbyte (no memory module plugged, resp.)</td>
</tr>
<tr>
<td>1 0 1</td>
<td>256 Mbyte</td>
</tr>
<tr>
<td>1 1 0</td>
<td>128 Mbyte</td>
</tr>
<tr>
<td>1 1 1</td>
<td>64 Mbyte</td>
</tr>
</tbody>
</table>

### 3.8.6 DRAM Organization

MEM-60 and on-board memory banks

The DRAM is mounted on-board or on the MEM-60 memory module. The on-board DRAM is arranged in 1 or 2 memory banks depending on the available overall memory capacity. Each memory bank is 36-bit wide – 32 data bits plus 4 parity bits.
1. FPM: Fast Page Mode

A parity bit checks every eight consecutive data bits (byte parity). The DRAM parity check is only performed when SW9-2 is set appropriately: OFF = enabled (default “OFF”, see page 13).

Table 29

<table>
<thead>
<tr>
<th>Product</th>
<th>DRAM device Type</th>
<th>Capacity</th>
<th>Total capacity</th>
<th>No. of banks</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU-60D/4</td>
<td>1M * 4 FPM</td>
<td>9 * 1 Mbit * 4</td>
<td>4 Mbyte</td>
<td>1</td>
</tr>
<tr>
<td>CPU-60D/8</td>
<td>1M * 4 FPM</td>
<td>18 * 1 Mbit * 4</td>
<td>8 Mbyte</td>
<td>2</td>
</tr>
<tr>
<td>CPU-60D/16</td>
<td>4M * 4 FPM</td>
<td>9 * 4 Mbit * 4</td>
<td>16 Mbyte</td>
<td>1</td>
</tr>
<tr>
<td>CPU-60D/32</td>
<td>4M * 4 FPM</td>
<td>18 * 4 Mbit * 4</td>
<td>32 Mbyte</td>
<td>2</td>
</tr>
</tbody>
</table>

1. FPM: Fast Page Mode

Bank selection

The bank selection depends on the number of installed memory banks:
- interleaved
  - The dual-banks architecture implements an interleaved memory organization of the DRAM: 4 consecutive bytes located in bank 1, the next 4 consecutive bytes in bank 2, etc.
- non-interleaved
  - The single-bank architecture implements a non-interleaved memory organization of the DRAM: every 4 consecutive bytes located in bank 1.

3.8.7 Cache Coherence and Snooping

To maintain cache coherence in a multimaster system, the 68060 CPU has the capability of snooping. On a snooped external bus cycle the 68060 CPU invalidates the cache line that is hit. Supplying dirty data and sinking dirty data is not supported by the 68060 CPU. Snoop hits invalidate the cache line in all cases (also for alternate master read/write cycles).

The snooping protocol supported by the 68060 CPU requires that memory areas shared with any other bus master is marked as ‘cacheable write-through’ or ‘cache inhibited’.
Unpredictable errors
Wrong configuration of snoop window and VMEbus slave window causes unpredictable errors. DRAM cache coherence can only be guaranteed if the snoop window in the bridge configuration register of the RIALTO bus bridge (see “SNOOP[2..0] (R/W)” on page 51) is configured correctly. Access to mirrored DRAM locations causes inconsistencies between the memory and the caches of the 68060 CPU.

- Never access mirrored DRAM locations.
- Ensure correct configuration: Usually the snoop window size equals the VMEbus slave window size. If not, round up the snoop window size to the next $2^x$-value so that the VMEbus slave window is entirely covered by the snoop window.
  - The snoop window in the bridge configuration register of the RIALTO bus bridge must be configured accordingly (see “SNOOP[2..0] (R/W)” on page 51).
  - The VMEbus slave window must be configured accordingly (see section 6.5.9 “INFO – Information about the CPU Board” on page 143 and the FORCE Gate Array FGA-002 User’s Manual).

### 3.8.8 DRAM Access from the 68060 CPU

**IMPORTANT**

After reset the boot PROM is mapped to address $0000.0000_{16}$. After initialization the firmware enables the DRAM at $0000.0000_{16}$ with an access to any of the 2 RIALTO registers (see table 3.6.1 “Register Set” on page 50).

<table>
<thead>
<tr>
<th>Start</th>
<th>End</th>
<th>Memory capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td>$0000.0000_{16}$</td>
<td>$01FF.FFFF_{16}$</td>
<td>32 Mbyte</td>
</tr>
<tr>
<td>$0000.0000_{16}$</td>
<td>$00FF.FFFF_{16}$</td>
<td>16 Mbyte</td>
</tr>
<tr>
<td>$0000.0000_{16}$</td>
<td>$007F.FFFF_{16}$</td>
<td>8 Mbyte</td>
</tr>
<tr>
<td>$0000.0000_{16}$</td>
<td>$003F.FFFF_{16}$</td>
<td>4 Mbyte</td>
</tr>
</tbody>
</table>
3.8.9 DRAM Access via the VMEbus

Shared RAM access via the VMEbus is routed by the FGA-002 (DRAM and user SRAM both contribute to shared RAM). The start and end access addresses are programmable in 4-Kbyte steps.

Write protection

The write protection of the programmed memory range depends on the VMEbus address modifier codes: For example, in privileged mode the memory can be read and written, while in non-privileged mode the memory can only be read, or a non-privileged access can be prohibited altogether.

Programmable access address range

The access address of the shared RAM for other VMEbus masters is programmable via the FGA-002. Both the start and the end address of the shared RAM are FGA-002 programmable in 4-Kbyte increments (see the FGA-002 Gate Array User's Manual). Therefore, the address range used by other VMEbus masters is not necessarily the same as the one used by the 68060 CPU for local accesses.

DRAM parity error

If a DRAM parity error is detected during a VMEbus slave read access, the memory controller terminates the cycle with an error acknowledge. Via the RIALTO bus bridge the error is signaled to the FGA-002, which drives the BERR signal. Thereby the parity error is signaled to the VMEbus master.

VMEbus access cycle

When the FGA-002 detects a VMEbus access cycle to the programmed address range of the shared RAM it requests bus mastership of the CPU bus via the RIALTO bus bridge from the CPU bus arbiter. After the arbiter has granted the CPU bus mastership to the FGA-002 the VMEbus access cycle is executed and all data is latched (read cycles) or stored to RAM (write cycles). After this the cycle is terminated and the FGA-002 immediately releases the local bus mastership back to the 68060 CPU. Simultaneously, it completes the fully asynchronous VMEbus access cycle.

Disabling early release

The early release of the memory read or write cycle allows the 68060 CPU to continue processing while the FGA-002 independently manages the VMEbus transaction overhead. The early bus release thereby enables early shared RAM accesses by the 68060 CPU, but sacrifices the guaranteed indivisibility of VMEbus read-modify-write shared RAM cycles (RMW).

Since the 68060 CPU includes an on-chip cache memory this may not effect the 68060 CPU performance at all but the bus bandwidth for uncached devices is broadened.

A programmable bit within the FGA-002 may be used to disable the early bus release option. When the early release is disabled, the FGA-002 retains the local bus mastership until the VMEbus cycle is finished. This guarantees that no other local bus master (68060 CPU or DMA control-
 ler) will access the shared RAM until the VMEbus cycle is completed. In case of a read-modify-write cycle (RMW) performed by another VME-
bus master, the FGA-002 will perform both transactions (a read followed
by a write) without releasing the local bus. This guarantees that the cycle
is indivisible.

3.8.10 DRAM Access from the Ethernet-Controller

The AM79C900 Ethernet controller uses DMA transfer cycles to transfer commands, data and status information to and from the DRAM.

3.8.11 DRAM Access from the SCSI-Controller

The SCSI 53C720SE uses DMA transfer cycles to run scripts and transfer data and status information to and from the DRAM.

3.9 User SRAM (factory option)

Backup 2 backup options are available to provide the current for the user SRAM standby mode (see section 3.9.1 “Backup Power for the User SRAM” on page 62).

Table 31 User SRAM features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data path width</td>
<td>32</td>
</tr>
<tr>
<td>Supported port size</td>
<td>Byte, word, long</td>
</tr>
<tr>
<td>Base address</td>
<td>Contiguous to DRAM</td>
</tr>
<tr>
<td>Number of devices</td>
<td>4</td>
</tr>
<tr>
<td>Location</td>
<td>J80…83</td>
</tr>
<tr>
<td>Supported device types</td>
<td>M5M5408L (512k * 8)</td>
</tr>
<tr>
<td>Default device speed</td>
<td>55 ns</td>
</tr>
</tbody>
</table>

User SRAM organization The user SRAM memory is connected to the memory bus, providing a long-wide port. Burst accesses are supported both for read and write. Data can be read from and written to any address; odd and even in byte, word, or long-word format.
Access address range  User SRAM and DRAM both contribute to shared RAM and shared RAM access via the VMEbus is routed by the FGA-002 (see section 3.8.9 “DRAM Access via the VMEbus” on page 60).

3.9.1 Backup Power for the User SRAM

The user SRAM is powered by the backup power circuitry.

Normal operation During normal operation the backup power circuitry connects the +5 V power supply to the user SRAM.

Power fail When the +5 V supply fails, backup power may be supplied from alternate sources. They are only available when SW5-3 is set appropriately: ON = RTC, local and user SRAM (default “OFF”, see page 11).

If SW5-3 is set appropriately, the following two alternate sources are switch-selectable:

- VME standby • from the VMEbus +5VSTDBY line; selectable by SW5-1: ON = enabled (default “OFF”, see page 11).
- Backup battery • from the backup battery; selectable by SW5-2: ON = enabled (default “OFF”, see page 11).

Automatic switch-over The switch-over in case of power fail is fully automatic; whichever voltage is higher will be available to the user SRAM.

3.10 System PROM

The system PROM consists of 4 flash memory devices. For the available capacity options see “System PROM” on page 2.

Memory organization The data path of the system flash memory is 32-bit wide. It is separated into 4 byte-paths, each byte-path is connected to one flash memory device.

Table 32 System PROM features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data path width</td>
<td>32-bit wide</td>
</tr>
<tr>
<td>Supported port size for read write</td>
<td>Long, Word, Byte Long (aligned)</td>
</tr>
<tr>
<td>Number of devices</td>
<td>4</td>
</tr>
</tbody>
</table>
3.10.1 Device Types for the System PROM

The following device types (or equivalent) are used as system PROM:

<table>
<thead>
<tr>
<th>Device type</th>
<th>Device speed</th>
<th>Total capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td>28F008SA: 1M * 8</td>
<td>85 ns</td>
<td>4 Mbyte</td>
</tr>
<tr>
<td>(Default configuration)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>29F016: 2M * 8</td>
<td>85 ns</td>
<td>8 Mbyte</td>
</tr>
</tbody>
</table>

3.10.2 Address Map of the System PROM

The base address of the system PROM is mapped via an address decoder and fixed to $\text{FF00.0000}_{16}$. The size of the address range depends on the memory capacity of the used devices.

<table>
<thead>
<tr>
<th>Start</th>
<th>End</th>
<th>Total capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{FF00.0000}_{16}$</td>
<td>$\text{FF3F.FFFF}_{16}$</td>
<td>4 Mbyte</td>
</tr>
<tr>
<td>$\text{FF00.0000}_{16}$</td>
<td>$\text{FF7F.FFFF}_{16}$</td>
<td>8 Mbyte</td>
</tr>
</tbody>
</table>
3.10.3 Reading and Programming the System PROM

**Reading**

Read cycles of any port size are allowed.

**Prerequisite for programming**

Programming the system PROM is only enabled when SW10-3 is set appropriately: OFF = writing enabled (default “OFF”, see page 13). The current setting of SW10-3 can be read from the WESYSFLASH bit in the MDR (see table 27 “Memory diagnostic register (MDR)” on page 55).

**Write termination**

Write burst cycles are terminated with a bus error by the memory controller.

**IMPORTANT**

- Before erasing or programming the system PROM ensure that you do not destroy the VMEPROM image. The VMEPROM image resides in the first 512 Kbyte of the system PROM starting at address FF00.0000₁₆ and ending at FF08.0000₁₆.
- All 4 devices can be programmed simultaneously. However, due to power consumption each device should be erased separately.

**Programming**

There are 2 more steps to be taken for programming the system PROM. Both steps are automatically handled correctly by the software packaged with the SYS68K/CPU-60 (see section 6.5.3 “FERASE – Erase Flash Memories” on page 137 and section 6.5.7 “FPROG – Program Flash Memories” on page 141).

1. A programming voltage $V_{PP}$ of 12 V must be applied to the flash devices making up the system PROM. $V_{PP}$ is generated by the SYS68K/CPU-60 and controlled via a register (see table 14 “CIO1 port B data register” on page 42). The $V_{PP}$ generator is shared between the system PROM, the user flash, and the boot PROM.

2. The device dependent communication sequence has to be performed on each of the 4 byte-paths.

3.11 Boot PROM

**Device selection**

The boot PROM devices are installed in two 32-pin PLCC sockets:

- $J70 =$ default
  - socket 1 (=J70) for the default boot PROM device
- $J71 =$ optional
  - and socket 2 (=J71) for the optional boot PROM device.

The selection of the boot PROM devices to be used is controlled by switch SW7-1 (default “OFF”, see page 12):

- OFF = Socket 1 – 0…512 Kbyte, Socket 2 – 512 Kbyte…1 Mbyte
- ON = Socket 1 disabled, Socket 2 from 0…1 Mbyte
Factory options

There are 3 device type factory options available (see section 3.11.1 “Boot PROM Address Map and Factory Options” on page 66):

- flash devices programmable at 12 V,
- flash devices programmable at 5 V,
- or OTP EPROM devices.

For the available capacity factory options see table 36 “Boot PROM address map, factory options, and device types” on page 66.

**IMPORTANT**

Ensure that there is always a boot PROM device providing a working boot PROM program installed.

Boot sequence

The 68060 CPU boots from the boot PROM after every power up or reset.
The boot PROM program boots up the 68060 CPU and initializes the FGA-002 register contents. During booting the FGA-002 maps all addresses to the boot PROM with the exception of the addresses of FGA-002 internal registers and the local SRAM.

Table 35 Boot PROM features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data path width</td>
<td>8-bit wide</td>
</tr>
<tr>
<td>Supported port size</td>
<td>Byte/word/long</td>
</tr>
<tr>
<td>Number of devices</td>
<td>1 or 2</td>
</tr>
<tr>
<td>Default number of devices</td>
<td>1 (default boot PROM is #1)</td>
</tr>
<tr>
<td>Default capacity</td>
<td>128 Kbyte</td>
</tr>
<tr>
<td>Default device type</td>
<td>28F010A, 12 V flash memory</td>
</tr>
<tr>
<td>Default device speed</td>
<td>100 ns</td>
</tr>
<tr>
<td>Default address range</td>
<td>FFE0.000016–FFE1.FFFF16</td>
</tr>
<tr>
<td>Forbidden function code on FLXI bus</td>
<td>111</td>
</tr>
</tbody>
</table>
3.11.1 Boot PROM Address Map and Factory Options

The base addresses of the default and the optional boot PROM are fixed that means they cannot be changed.

- After booting one of the boot PROM devices is accessible at base address $\text{FFE}0.0000_{16}$, regardless of the SW7-1 setting:
  - If SW7-1 is in its default setting (default “OFF”, see page 12), $\text{FFE}0.0000_{16}$ is the base address of the default PROM device in socket 1.
  - Otherwise, $\text{FFE}0.0000_{16}$ is the base address of the optional PROM device in socket 2.

- If the optional boot PROM device is installed and if socket 1 is not disabled by SW7-1 being set to ON, the optional boot PROM device is accessible at $\text{FFE}8.0000_{16}$.

**IMPORTANT**

After reset, the boot PROM is mapped to address $0000.0000_{16}$. After initialization the firmware enables the DRAM at $0000.0000_{16}$ with an access to any of the RIALTO registers.

Factory options

The following factory options are available for the boot PROM using the listed device types (or equivalent):

<table>
<thead>
<tr>
<th>Table 36</th>
<th>Boot PROM address map, factory options, and device types</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Factory option</th>
<th>Device type</th>
<th>Offset range for each installed device (base address as documented above)</th>
<th>Total capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td>12V 28F010A: 128k * 8 12 V flash memory, only in socket 1 – default –</td>
<td>$0.0000_{16}$–$1.FFFF_{16}$</td>
<td>128 Kbyte</td>
<td></td>
</tr>
<tr>
<td>28F020A: 256k * 8 12 V flash memory only in socket 1</td>
<td>$0.0000_{16}$–$3.FFFF_{16}$</td>
<td>256 Kbyte</td>
<td></td>
</tr>
<tr>
<td>28F020A: 256k * 8 12 V flash memory both in socket 1 and 2</td>
<td>$0.0000_{16}$–$7.FFFF_{16}$</td>
<td>512 Kbyte</td>
<td></td>
</tr>
<tr>
<td>5V 29F040: 512k * 8 5 V flash memory, both in socket 1 and 2</td>
<td>$0.0000_{16}$–$F.FFFF_{16}$</td>
<td>1 Mbyte</td>
<td></td>
</tr>
</tbody>
</table>
Table 36  Boot PROM address map, factory options, and device types (cont.)

<table>
<thead>
<tr>
<th>Factory option</th>
<th>Device type</th>
<th>Offset range for each installed device (base address as documented above)</th>
<th>Total capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td>OTP</td>
<td>27C040: 512k * 8 OTP EPROM, both in socket 1 and 2</td>
<td>0.0000₁₆–FFFF₁₆</td>
<td>1 Mbyte</td>
</tr>
<tr>
<td>Supported device types, but not available as factory option</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Offset range for each installed device (base address as documented above)</th>
<th>Total capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td>12V 28F512A: 64k * 8 12 V flash memory</td>
<td>n/a</td>
</tr>
<tr>
<td>28C512: 64k * 8 12 V flash memory</td>
<td>n/a</td>
</tr>
<tr>
<td>28C010: 128k * 8 12 V flash memory</td>
<td>n/a</td>
</tr>
<tr>
<td>5V 29F010: 128k * 8 5 V flash memory</td>
<td>n/a</td>
</tr>
<tr>
<td>OTP 27C010: 128k * 8 OTP EPROM</td>
<td>n/a</td>
</tr>
<tr>
<td>27C020: 256k * 8 OTP EPROM</td>
<td>n/a</td>
</tr>
<tr>
<td>27C080: 1M * 8 OTP EPROM</td>
<td>n/a</td>
</tr>
</tbody>
</table>

3.11.2  Programming the Boot PROM

Writing to the boot PROM is only enabled

- when using flash memory devices
- and when SW7-4 is set appropriately: ON = writing enabled (default “OFF”, see page 12).

**IMPORTANT**  Before erasing or programming the boot PROM ensure that you do not destroy the FORCE COMPUTERS FGA Boot image. Before erasing or programming make a copy of the boot PROM device 1 in socket J70.

After enabling programming there is 1 more step to be taken for programming the boot PROM. The step is automatically handled correctly by the software packaged with the SYS68K/CPU-60 (see section 6.5.3 “FE-
RASE – Erase Flash Memories” on page 137 and section 6.5.7 “FPROG – Program Flash Memories” on page 141) and by the assembly process.

- The correct programming voltage $V_{pp}$ must be applied to the flash devices making up the boot PROM. $V_{pp}$ is generated by the SYS68K/CPU-60 and controlled via a register (see table 14 “CIO1 port B data register” on page 42). The $V_{pp}$ generator is shared between the system PROM, the user flash, and the boot PROM.

### 3.12 User Flash

The user flash is a user programmable flash device.

**Location**

J31

**Base address**

$\text{FFC8.0000}_{16}$

**Device type factory options**

There are 2 device type factory options available

- flash devices programmable at 12 V
- and flash devices programmable at 5 V.

The following factory options are available for the user flash using the device types listed (or equivalent): 

<table>
<thead>
<tr>
<th>Factory option</th>
<th>Device type</th>
<th>Address range</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. 28F020:256k * 8</td>
<td>12 V flash memory</td>
<td>$\text{FFC8.0000}<em>{16}$-$\text{FFCB.FFFF}</em>{16}$</td>
</tr>
<tr>
<td>2. 28F010:128k * 8</td>
<td>12 V flash memory</td>
<td>$\text{FFC8.0000}<em>{16}$-$\text{FFC9.FFFF}</em>{16}$</td>
</tr>
<tr>
<td>3. 29F040:512k * 8</td>
<td>5V flash memory</td>
<td>$\text{FFC8.0000}<em>{16}$-$\text{FFCF.FFFF}</em>{16}$</td>
</tr>
</tbody>
</table>

### 3.12.1 Programming the User Flash

Writing to the user flash is only enabled when SW10-4 is set appropriately: OFF = writing enabled (default “OFF”, see page 12).

After enabling programming there is 1 more step to be taken for programming the user flash. The step is automatically handled correctly by the
software packaged with the SYS68K/CPU-60 (see section 6.5.3 “FREASURE – Erase Flash Memories” on page 137 and section 6.5.7 “FPROG – Program Flash Memories” on page 141) and by the assembly process.

- The correct programming voltage $V_{PP}$ must be applied to the flash devices making up the boot PROM. $V_{PP}$ is generated by the SYS68K/CPU-60 and controlled via a register (see table 14 “CIO1 port B data register” on page 42). The $V_{PP}$ generator is shared between the system PROM, the user flash, and the boot PROM.

### 3.13 Local SRAM

<table>
<thead>
<tr>
<th>Location</th>
<th>J51</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base address</td>
<td>FFC0.0000&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>Backup</td>
<td>2 backup options are available to provide the current for the local SRAM standby mode (see section 3.13.3 “Backup Power for the Local SRAM” on page 71).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 38</th>
<th>Local SRAM features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feature</td>
<td>Value</td>
</tr>
<tr>
<td>Data path width</td>
<td>Byte</td>
</tr>
<tr>
<td>Supported port size</td>
<td>Byte, word, long</td>
</tr>
<tr>
<td>Number of devices</td>
<td>1</td>
</tr>
<tr>
<td>Default number of devices</td>
<td>1</td>
</tr>
<tr>
<td>Default capacity</td>
<td>128 Kbyte</td>
</tr>
<tr>
<td>Default device type</td>
<td>M5M 510008L</td>
</tr>
<tr>
<td>Default device speed</td>
<td>100 ns</td>
</tr>
<tr>
<td>Default address range</td>
<td>FFC0.0000&lt;sub&gt;16&lt;/sub&gt;–FFC1.FFFF&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>Forbidden function code on FLXI bus</td>
<td>111</td>
</tr>
</tbody>
</table>

#### 3.13.1 Local SRAM Organization

The local SRAM memory is connected to the I/O bus, providing a byte-wide port. Consecutive bytes seen by the 68060 CPU are handled in the same manner as consecutive bytes for the local SRAM.
Byte, word, and long word accesses are managed by the dynamic bus sizing of the RIALTO bus bridge (see section 3.6 “RIALTO Bus Bridge” on page 50).
Data can be read from and written to any address; odd and even in byte, word, or long word format.

Example of data transfers:

All combinations of the instructions listed below are allowed:

```
MOVE.X ($FFC0 000Y), D0
```

\[X = B = \text{Byte} \quad 1 \text{ Byte}\]
\[X = W = \text{Word} \quad 2 \text{ Bytes}\]
\[X = L = \text{Long Word} \quad 4 \text{ Bytes}\]

\[Y = 0\]
\[Y = 1\]
\[Y = 2\]
\[Y = 3\]


3.13.2 Devices Types for the Local SRAM

The following low power device types (marked with -L or -LL) are supported as a factory option.

Table 39 Local SRAM factory options and device types

<table>
<thead>
<tr>
<th>Factory option (Default configuration is the first option listed)</th>
<th>Device type</th>
<th>Address range</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. M5M 510008L: 128k * 8</td>
<td>FFC0.0000\text{16}...FFC1.FFFF\text{16}</td>
<td></td>
</tr>
<tr>
<td>2. M5M5256L: 32k * 8</td>
<td>FFC0.0000\text{16}...FFCO.7FFF\text{16}</td>
<td></td>
</tr>
<tr>
<td>3. M5M5408L: 512k * 8</td>
<td>FFC0.0000\text{16}...FFC7.FFFF\text{16}</td>
<td></td>
</tr>
</tbody>
</table>
3.13.3 Backup Power for the Local SRAM

The local SRAM is powered by the backup power circuitry.

<table>
<thead>
<tr>
<th>Normal operation</th>
<th>During normal operation the backup power circuitry connects the +5 V power supply to the local SRAM.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power fail</td>
<td>When the +5 V supply fails, backup power may be supplied from alternate sources. They are only available with SW5-3 set appropriately: ON = RTC, local and user SRAM (default “OFF”, see page 11).</td>
</tr>
</tbody>
</table>

If SW5-3 is set appropriately, the following two alternate sources are switch-selectable:

- **VME standby** • from the VMEbus \(+5\text{VSTDBY}\) line; selectable by SW5-1: ON = enabled (default “OFF”, see page 11).
- **Backup battery** • from the backup battery; selectable by SW5-2: ON = enabled (default “OFF”, see page 11).

**Automatic switch-over** The switch-over in case of power fail is fully automatic; whichever voltage is higher will be available to the local SRAM.

3.14 Real-Time Clock – RTC 72423

The on-board RTC 72423 maintains accurate time and date based on its own crystal.

**Backup** 2 backup options are available to provide the current for the RTC 72423 even during power-failures (see section 3.14.3 “Backup Power for the RTC 72423” on page 74).

**Data sheet** See data sheet “RTC 72421” in section 5.

**Table 40** RTC 72423 features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supported port size</td>
<td>Byte only (D3…0 valid)</td>
</tr>
<tr>
<td>Access mode</td>
<td>Byte only</td>
</tr>
<tr>
<td>Access address</td>
<td>( FF80.3000_{16} )</td>
</tr>
</tbody>
</table>
3.14.1 RTC Registers Address Map

The RTC 72423 has a 4-bit data bus which has to be accessed in byte mode. The upper four bits (4..7) are "don’t care" during read and write accesses.

Base address  \(\text{FF80.3000}_{16}\)

Table 41 RTC registers address map

<table>
<thead>
<tr>
<th>Offset</th>
<th>Register name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(00_{16})</td>
<td>RTC1SEC</td>
<td>– 1 second digit reg.</td>
</tr>
<tr>
<td>(01_{16})</td>
<td>RTC10SEC</td>
<td>– 10 second digit reg.</td>
</tr>
<tr>
<td>(02_{16})</td>
<td>RTC1MIN</td>
<td>– 1 minute digit reg.</td>
</tr>
<tr>
<td>(03_{16})</td>
<td>RTC10MIN</td>
<td>– 10 minute digit reg.</td>
</tr>
<tr>
<td>(04_{16})</td>
<td>RTC1HR</td>
<td>– 1 hour digit reg.</td>
</tr>
<tr>
<td>(05_{16})</td>
<td>RTC10HR</td>
<td>– PM/AM and 10 hour digit reg.</td>
</tr>
<tr>
<td>(06_{16})</td>
<td>RTC1DAY</td>
<td>– 1 day digit reg.</td>
</tr>
<tr>
<td>(07_{16})</td>
<td>RTC10DAY</td>
<td>– 10 day digit reg.</td>
</tr>
<tr>
<td>(08_{16})</td>
<td>RTC1MON</td>
<td>– 1 month digit reg.</td>
</tr>
<tr>
<td>(09_{16})</td>
<td>RTC10MON</td>
<td>– 10 month digit reg.</td>
</tr>
<tr>
<td>(0A_{16})</td>
<td>RTC1YR</td>
<td>– 1 year digit reg.</td>
</tr>
<tr>
<td>(0B_{16})</td>
<td>RTC10YR</td>
<td>– 10 year digit reg.</td>
</tr>
<tr>
<td>(0C_{16})</td>
<td>RTCWEEK</td>
<td>– Week reg.</td>
</tr>
<tr>
<td>(0D_{16})</td>
<td>RTCCOND</td>
<td>– Control reg. D</td>
</tr>
<tr>
<td>(0E_{16})</td>
<td>RTCCONE</td>
<td>– Control reg. E</td>
</tr>
<tr>
<td>(0F_{16})</td>
<td>RTCCOF</td>
<td>– Control reg. F</td>
</tr>
</tbody>
</table>
3.14.2 Reading from or Writing to the RTC 72423

**IMPORTANT** Stop the RTC 72423 before reading the date and time registers.

Example:

The following programming example shows how to read from or write to the RTC 72423.

```c
/*****************************************
** read RTC 72423 and load to RAM     **
** 30-Oct-87  M.S.                  **
*****************************************/

setclock(sy)
register struct SYRAM *sy;
{
    register struct rtc7242 *rtc = RTC2;
    register long count=1000001;

    rtc->dcontrol = 1; /* hold clock */
    while(--count)
        if(rtc->dcontrol&0x02)
            break;
    if(!count)
        { printf("\nCannot read Realtime Clock");
            rtc->dcontrol = 0;
            return; }
    sy->_ssec[0] = (unsigned char)((rtc->sec10reg&0x07)*10 + (rtc->sec1reg&0x0f));
    sy->_smin = (unsigned char)((rtc->min10reg&0x07)*10 + (rtc->min1reg&0x0f));
    sy->_shrs = (unsigned char)((rtc->hou10reg&0x03)*10 + (rtc->hou1reg&0x0f));
    sy->_syrs[0] = (unsigned char)((rtc->yr10reg&0x0f) *10 + (rtc->yr1reg&0x0f));
    sy->_sday = (unsigned char)((rtc->day10reg&0x03)*10 + (rtc->day1reg&0x0f));
    sy->_smon = (unsigned char)((rtc->mon10reg&0x01)*10 + (rtc->mon1reg&0x0f));
    rtc->dcontrol = 0; /* start clock */
}
```
3.14.3  Backup Power for the RTC 72423

The RTC 72423 is powered by the backup power circuitry.

Normal operation
During normal operation the backup power circuitry connects the +5 V
power supply to the RTC 72423.

Power fail
When the +5 V supply fails, backup power may be supplied from alter-
nate sources:

– VME standby
  • from the VMEbus +5VSTDBY line; selectable by SW5-1: ON =
    enabled (default “OFF”, see page 11).

– Backup battery
  • from the backup battery; selectable by SW5-2: ON = enabled (default
    “OFF”, see page 11).

Automatic
switch-over
The switch-over in case of power fail is fully automatic; whichever volt-
age is higher will be available to the user SRAM.
3.15 VMEbus Interface

The following sections describe the VMEbus interface in detail. This section gives a short overview of the VMEbus interface features.

- **ANSI/VITA compliance**
  
  The SYS68K/CPU-60 provides a complete VMEbus interface compliant with ANSI/VITA 1-1994.

- **Supported transfers**
  
  The VMEbus interface supports 8, 16, and 32 bit, as well as unaligned data transfers. The extended, standard, and short I/O address modifier codes are implemented to interface the SYS68K/CPU-60 to all existing VMEbus products.

- **RMW cycles**
  
  Read-modify-write cycles on the VMEbus (RMW cycles) are also supported. The address strobe signal is held low during RMW cycles while the data strobe signals are driven low twice, once for the read cycle and once for the write cycle, and high between both of them.

- **Interrupt handler**
  
  The complete VMEbus interrupt management is done by the FGA-002 enabling the use of a high-end multiprocessor environment board with distributed interrupt handling. The FGA-002 acts as D08(O) interrupt handler in compliance with the VMEbus specification. 16-bit interrupt vectors are not supported.

  All 7 VMEbus interrupt request (IRQ) signals are connected to the interrupt handling logic on the FGA-002.
  
  - All 7 VMEbus IRQ signals can be separately enabled or disabled.
  
  - Every VMEbus interrupt request level can be mapped to cause an interrupt to the processor on a different level. For example, a VMEbus interrupt request on level 2 (IRQ2*) can be mapped to cause an interrupt request to the 68060 CPU on level 5.

- **Slot-1**
  
  A single-level bus arbiter together with several release functions is implemented with all slot-1 system controller functions (see section 3.17 “VMEbus Slot-1” on page 86):
  
  - SYSRESET* driver and receiver,
  
  - SYSCLK driver,
  
  - and IACK daisy-chain driver (see below).

- **IACK Daisy Chain Driver**
  
  In accordance with the VMEbus specification the CPU board includes an IACK daisy-chain driver. If the CPU board is plugged in slot 1 and configured accordingly by SW6-1 and SW6-2, the board acts as IACK daisy-chain driver. Plugged in any other slot the board closes the IACKIN-IACKOUT path.
**NOTICE**

Damaging SYS68K/CPU-60 components

On the backplane the jumper for IACKIN-IACKOUT-bypass must be removed for proper operation. This is not necessary on active backplanes.

**IOBP-1**

For the connections to the SYS68K/IOBP-1 I/O panel see section 2.12 “SYS68K/IOBP-1” on page 28.

### 3.15.1 Exception Signals SYSFAIL, SYSRESET, and ACFAIL

The VMEbus specification includes the signals SYSFAIL*, SYSRESET* and ACFAIL* for signalling exceptions or status. The SYSFAIL*, SYSRESET* and ACFAIL* signals are connected to the CPU board via buffers, switches, and the FGA-002.

**SYSFAIL***

The FGA-002 may be programmed to generate local interrupts when the SYSFAIL* signal is active. The VMEPROM firmware monitors the SYSFAIL* line during the initialization of external intelligent I/O boards.

**SYSRESET***

- **input**
  
  The VMEbus SYSRESET* signal is only monitored by the CPU board if SW9-4 is set appropriately: OFF = enabled (default “OFF”, see page 13).

- **output**
  
  A SYSRESET* is generated by the SYS68K/CPU-60 for any one of the following reasons:
  
  - the front panel reset key is active,
  - a RESET instruction is executed by the 68060 CPU on the local bus,
  - the FGA-002 reset register is accessed,
  - the watchdog timer is reset,
  - power-up occurs,
  - or the voltage monitor detects a low voltage condition on-board.

  The SYSRESET* signal is only passed to the VMEbus if SW9-3 is set appropriately: OFF = enabled (default “OFF”, see page 13).

**ACFAIL***

The ACFAIL* line is ignored by VMEPROM. The VMEbus requester logic in the FGA-002 monitors the ACFAIL* signal and may force a release of the VMEbus mastership when ACFAIL* is asserted. The CPU board can never drive the ACFAIL* signal.
3.15.2 Master Interface: Address Modifier (AM) Codes

The VMEbus defines 3 different address modifier ranges as shown in the following table:

<table>
<thead>
<tr>
<th>Name</th>
<th>Address lines used</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A32</td>
<td>A1…31</td>
<td>Extended addressing</td>
</tr>
<tr>
<td>A24</td>
<td>A1…23</td>
<td>Standard addressing</td>
</tr>
<tr>
<td>A16</td>
<td>A1…15</td>
<td>Short I/O</td>
</tr>
</tbody>
</table>

The 4-Gbyte address range of the 68060 CPU is split into address ranges to support all AM codes listed in the table below. Additionally, the table lists the AM codes which the SYS68K/CPU-60 drives and relates them to the address ranges.

**IMPORTANT** All VMEbus slave boards which are to be addressed by the SYS68K/CPU-60 must recognize one or more of the AM codes in the following table to guarantee proper operation.

Abbreviations The abbreviations below will be used in the following table:

- SPA Supervisor Program Access
- SDA Supervisor Data Access
- NPA Non-Privileged Program Access
- NDA Non-Privileged Data Access

<table>
<thead>
<tr>
<th>Address range</th>
<th>AM code</th>
<th>Code</th>
<th>Address and data bus width</th>
</tr>
</thead>
<tbody>
<tr>
<td>0xx0.000016...F9FF.FFFF16</td>
<td>0E16</td>
<td>SPA</td>
<td>VMEbus extended access</td>
</tr>
<tr>
<td></td>
<td>0D16</td>
<td>SDA</td>
<td>A32: D32/D24/D16/D8</td>
</tr>
<tr>
<td></td>
<td>0A16</td>
<td>NPA</td>
<td>(xx depending on shared memory)</td>
</tr>
<tr>
<td></td>
<td>0916</td>
<td>NDA</td>
<td>VMEbus standard access</td>
</tr>
<tr>
<td>FA00.000016...FAFF.FFFF16</td>
<td>0D16</td>
<td>SDA</td>
<td>FORCE message broadcast range</td>
</tr>
<tr>
<td></td>
<td>0916</td>
<td>NDA</td>
<td></td>
</tr>
<tr>
<td>FB00.000016...FBFE.FFFF16</td>
<td>3E16</td>
<td>SPA</td>
<td>VMEbus extended access</td>
</tr>
<tr>
<td></td>
<td>3D16</td>
<td>SDA</td>
<td>A24: D32/D24/D16/D8</td>
</tr>
<tr>
<td></td>
<td>3A16</td>
<td>NPA</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3916</td>
<td>NDA</td>
<td>VMEbus standard access</td>
</tr>
</tbody>
</table>
### 3.15.3 Master Interface: Data Transfer Size

**Fixed and programmable D32/D16**

The VMEbus address range is the largest portion of the memory map (see section 3.1 “SYS68K/CPU-60 Memory Map” on page 36). It is divided into ranges with address and data bus widths varying between different ranges but fixed within a range (A32/A24/A16 and D32/D16, respectively). The VMEbus master interface also contains address ranges where the data transfer size is software programmable to be 16-bit or 32-bit wide.

**Automatic 32-to-16-bit transformation**

When the data transfer bus width for an address range is limited to 16 bit and a 32-bit transfer is attempted, the CPU board hardware will automatically perform two consecutive transfers, so that no software overhead is necessary.

The following table lists the VMEbus address ranges and their associated address and data bus widths in detail.

<table>
<thead>
<tr>
<th>Table 43</th>
<th>Address ranges related to AM codes (cont.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address range</td>
<td>AM code</td>
</tr>
<tr>
<td>FBFF.0000_16...FBFF.FFFF_16</td>
<td>2D&lt;sub&gt;16&lt;/sub&gt; 10.1101&lt;sub&gt;2&lt;/sub&gt; 10.1001&lt;sub&gt;2&lt;/sub&gt;</td>
</tr>
<tr>
<td>FCO0.0000_16...FCFE.FFFF_16</td>
<td>3E&lt;sub&gt;16&lt;/sub&gt; 11.1110&lt;sub&gt;2&lt;/sub&gt; 3D&lt;sub&gt;16&lt;/sub&gt; 11.1101&lt;sub&gt;2&lt;/sub&gt; 3A&lt;sub&gt;16&lt;/sub&gt; 11.1010&lt;sub&gt;2&lt;/sub&gt; 39&lt;sub&gt;16&lt;/sub&gt; 11.1001&lt;sub&gt;2&lt;/sub&gt;</td>
</tr>
<tr>
<td>FCPF.0000_16...FCFF.FFFF_16</td>
<td>2D&lt;sub&gt;16&lt;/sub&gt; 10.1101&lt;sub&gt;2&lt;/sub&gt; 29&lt;sub&gt;16&lt;/sub&gt; 10.1001&lt;sub&gt;2&lt;/sub&gt;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 44</th>
<th>Bus widths related to address ranges: VMEbus master interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start</td>
<td>End</td>
</tr>
<tr>
<td>xxxx.0000&lt;sub&gt;16&lt;/sub&gt;</td>
<td>FAFF.FFFF&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>xxxx depends on the shared RAM capacity.</td>
<td></td>
</tr>
<tr>
<td>FB00.0000&lt;sub&gt;16&lt;/sub&gt;</td>
<td>FBFE.FFFF&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>FBFF.0000&lt;sub&gt;16&lt;/sub&gt;</td>
<td>FF FF&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>FCO0.0000&lt;sub&gt;16&lt;/sub&gt;</td>
<td>FCFE.FFFF&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>FCFF.0000&lt;sub&gt;16&lt;/sub&gt;</td>
<td>FCFF.FFFF&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
</tbody>
</table>
For further information on snooping and read-modify-write support, see section 3.8.7 “Cache Coherence and Snooping” on page 58 and section 3.8.9 “DRAM Access via the VMEbus” on page 60.

**VMEPROM**

VMEPROM automatically reads the setting of the front-panel rotary switches to select the data bus size of the VMEbus after reset or power up (see section 6.2.3 “Rotary Switches” on page 128). Thereby, VMEPROM allows easy installation of additional memory boards with known data sizes during user program or operating system start. Additionally, the VMEPROM MEM command can be used to set up the data bus transfer size of the programmable address ranges (see section 6.5.10 “MEM – Set Data Bus Width of the VMEbus” on page 143).

**Table 45**

VMEbus master transfer cycles defined for D32 data bus width

<table>
<thead>
<tr>
<th>Transfer type</th>
<th>D31 ...24</th>
<th>D23 ...16</th>
<th>D15 ...08</th>
<th>D07 ...00</th>
</tr>
</thead>
<tbody>
<tr>
<td>Byte on odd address</td>
<td></td>
<td></td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>Byte on even address</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>Word</td>
<td></td>
<td></td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>Long-word</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>Unaligned word</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Unaligned long-word A</td>
<td></td>
<td>x</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>Unaligned long-word B</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>Read-modify-write</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>byte on odd address</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>byte on even address</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>word</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>long-word</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
</tbody>
</table>

**Table 46**

VMEbus master transfer cycles defined for D16 data bus width

<table>
<thead>
<tr>
<th>Transfer type</th>
<th>D31 ...24</th>
<th>D23 ...16</th>
<th>D15 ...08</th>
<th>D07 ...00</th>
</tr>
</thead>
<tbody>
<tr>
<td>Byte on odd address</td>
<td></td>
<td></td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>Byte on even address</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>Word</td>
<td></td>
<td></td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>Read-modify-write</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>byte on odd address</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>byte on even address</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>word</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
</tbody>
</table>

**Table 47**

VMEbus master transfer cycles defined for D8 data bus width

<table>
<thead>
<tr>
<th>Transfer type</th>
<th>D31 ...24</th>
<th>D23 ...16</th>
<th>D15 ...08</th>
<th>D07 ...00</th>
</tr>
</thead>
<tbody>
<tr>
<td>Byte on odd address</td>
<td></td>
<td></td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>Byte on even address</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>Word</td>
<td></td>
<td></td>
<td>x</td>
<td></td>
</tr>
</tbody>
</table>
3.15.4 Master Interface: Burst to VMEbus

On the initial cycle of a line transfer, a retry causes the 68060 CPU to retry the bus cycle. Contrasting to this, a retry signaled during the 2nd, 3rd, or 4th cycle of a line transfer is recognized as a bus error, and causes the CPU to abort the line transfer and start an access fault exception subroutine.

This different behaviour results in a different behaviour when encountering bus collisions during the 1st or during the 2nd, 3rd, or 4th cycle of a line transfer:

- **Bus collision during 1st cycle**
  - When the 68060 CPU wants to access a slave on the VMEbus and has already been granted the local bus
  - and when a master on the VMEbus wants to access the SYS68K/CPU-60’s shared RAM and has already been granted the VMEbus,

  a bus collision occurs. In this case, the FGA-002 signals a retry to the 68060 CPU to resolve the collision on the hardware level. Therefore, it is not necessary for the software to observe this event.

- **Bus collision during 2nd to 4th cycle**
  Opposite to the situation just described, the 68060 CPU initiates a bus error when a bus collision occurs during the 2nd, 3rd, or 4th cycle of a line transfer where the CPU is not able to retry the cycle. So the collision appears on the software level and can be resolved there but only with considerable time expense.

To prevent the software from being concerned, the SYS68K/CPU-60 implements the following feature:

- **Solution: locked RMC transfer**
  A line transfer from the 68060 CPU is defined as a locked RMC (read-modify-cycle) transfer on the FLXI bus. So the FGA-002, when being granted the VMEbus, does not release the VMEbus until all 4 long cycles of the line transfer are successfully completed or an actual bus error occurred.

- **To use this feature**
  When using this feature the URMW bit (= bit 7) of the FGA-002 CTL16 register has to be set to 1 (VMEPROM sets it to 0).
  - Thereby, line transfers to a D16-slave are enabled.
  - Additionally, the FGA-002 thereby is programmed to release ASVME high between the locked RMC similar transfers and not to support real VMEbus compatible RMCs. Actual RMC transfers from the 68060 CPU are treated the same way. As a result, this kind of arbitration locked RMC can be broken on a slave board which is accessible from the VMEbus and from the VME secondary bus.
3.15.5 Slave Interface: Access Address

The access address of the shared RAM for other VMEbus masters is programmable via the FGA-002. Both the start and the end address of the shared RAM are FGA-002 programmable in 4-Kbyte increments (see the FORCE Gate Array FGA-002 User’s Manual).

3.15.6 Slave Interface: DRAM Data Transfer Size

The VMEbus slave interface for the shared RAM is 32-bit wide. It supports 32-bit, 16-bit, and 8-bit as well as unaligned (UAT) and read-modify-write transfers.

3.15.7 Slave Interface: Address Modifier Decoding and A24 Slave Mode

For VMEbus slave access to the shared RAM, extended (A32) and standard (A24) accesses are allowed.

The on-board logic allows accesses in the privileged (supervisor) or non-privileged (user) mode for both data and program accesses. Each access mode can be separately enabled or disabled within the FGA-002.

Example:

Read and write permission can be enabled for supervisor accesses, and read permission for user accesses.

Automatic A24-to-A32 translation

Although A32 and A24 accesses are allowed, the FGA-002 only recognizes A32 accesses. If an A24 access occurs and the CIO2 is configured appropriately (see section 3.3.4 “On-board DRAM Capacity and Automatic A24 Expansion” on page 44), additional hardware automatically translates the A24 access to an A32 access to the FGA-002. This means that the standard address modifier code from the VMEbus is automatically modified to extended address modifier to the FGA-002. Since during A24 accesses the address lines A31…24 of the VMEbus must not be used for address decoding these address lines are driven to the FGA-002 via an additional driver. The value of the A31…24 bits are programmable (see section 3.3.6 “A24-to-A32 Address Translation” on page 45). The address lines for the A31…24 bits must be programmed according to the actual A32 access address used by the FGA-002.
**Example:**

Suppose the DRAM access address for the VMEbus A32 slave window is programmed to:

- Start address: \(1000.0000_{16}\)
- End address: \(1040.0000_{16}\)

Then the CIO2 Port A register must be programmed to \(10_{16}\) to allow A24 accesses (see section Table 18 “CIO2 port A data register” on page 45). If an A24 master now accesses the address \(00.5000_{16}\), it reaches the same address as an A32 master accessing the address \(1000.5000_{16}\).

**Programming the A32 access address**

For information on programming the A32 access address, see the *FGA-002 Gate Array User’s Manual*. The snooping window must be set appropriately (see section 3.8.7 “Cache Coherence and Snooping” on page 58).

**Enabled modes**

The A32 mode is always enabled and the A24 mode can be enabled additionally (see section 3.3.4 “On-board DRAM Capacity and Automatic A24 Expansion” on page 44 and "System Flags" in section 8.2.12 “SET-UP – Change Initialization Values” on page 193).

The following table shows the allowed AM codes for VMEbus accesses to the DRAM.

**Table 47 VMEbus slave AM codes**

<table>
<thead>
<tr>
<th>AM Code</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>3E(_{16})</td>
<td>11.1110(_2)</td>
</tr>
<tr>
<td>3D(_{16})</td>
<td>11.1101(_2)</td>
</tr>
<tr>
<td>3A(_{16})</td>
<td>11.1010(_2)</td>
</tr>
<tr>
<td>39(_{16})</td>
<td>11.1001(_2)</td>
</tr>
<tr>
<td>0E(_{16})</td>
<td>00.1110(_2)</td>
</tr>
<tr>
<td>0D(_{16})</td>
<td>00.1101(_2)</td>
</tr>
<tr>
<td>0A(_{16})</td>
<td>00.1010(_2)</td>
</tr>
<tr>
<td>09(_{16})</td>
<td>00.1001(_2)</td>
</tr>
</tbody>
</table>

**3.15.8 Slave Interface: Locked Cycles**

To support RMW-cycles for slave accesses the `SHAREDRMW` bit (= bit 0) of the FGA-002 CTL15 register has to be set to 1 (VMEPROM sets it to 0).
3.16 VMEbus Arbitration

Each transfer to or from an off-board address causes a VMEbus access cycle. The VMEbus defines an arbitration mechanism to arbitrate for bus mastership. The CPU board includes

- a VMEbus arbiter so that it may act as slot-1 system controller;
- a VMEbus requester so that it may access external VMEbus resources.

3.16.1 Single-Level VMEbus Arbiter

The CPU board contains a single level arbiter which can be enabled or disabled by software (see the FORCE Gate Array FGA-002 User’s Manual). No additional control of the arbiter is required.

**IMPORTANT**

- The arbiter of the FGA-002 will not be set automatically by hardware when detecting slot-1 by switch setting or auto-detection. It must be enabled by software if the CPU board is system controller (e.g., FGA Boot enables the arbiter automatically). For more information on the FGA-002 arbiter, please see the FORCE Gate Array FGA-002 User’s Manual.

- In accordance with the VMEbus specification, the arbiter must be enabled if the CPU board is located in the slot 1 of the VMEbus backplane. It must be disabled if the CPU board is located in any other slot.

- When the on-board single-level VMEbus arbiter is enabled, all other VMEbus masters (if any) must request VMEbus mastership using only bus request level 3 (BR3* signal). Otherwise, they are not recognized by the SYS68K/CPU-60.

3.16.2 VMEbus Requester

The SYS68K/CPU-60 includes a VMEbus requester so that it may access external VMEbus resources.

**Request (arbitration) level selection**

The request level is either selected automatically or by switch setting:

- If the SYS68K/CPU-60 detects slot 1, the request level 3 will automatically be used.

- If the SYS68K/CPU-60 does not detect slot 1, the request level is switch selectable (by SW6-3 and SW6-4: default “OFF OFF = level 3 (BR3*)”, see page 12).

For a detailed description of the slot-1 detection, see section 3.17 “VMEbus Slot-1” on page 86.
3.16.3 VMEbus Release Modes

The CPU board provides several software-selectable VMEbus release modes to release VMEbus mastership. The bus release operation is independent of the fact whether the on-board VMEbus arbiter is enabled and independent of the VMEbus arbitration level. Easy handling and use of the VMEbus release modes is provided by the FGA-002. Before the bus is released a read-modify-write (RMW) cycle in progress is always completed.

VMEPROM The VMEPROM `ARB` command sets the VMEbus release modes (see section 6.5.1 “ARB – Set the Arbiter of the CPU Board” on page 136).

Each row of the following table lists which of the VMEbus release modes described below can be used simultaneously (ROR and RAT are always enabled):

<table>
<thead>
<tr>
<th>Table 48</th>
<th>Valid configurations for VMEbus release modes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Config.</td>
<td>Release mode</td>
</tr>
<tr>
<td>1. REC, ROR, RAT, RBCLR</td>
<td>Yes Always Always Don’t care</td>
</tr>
<tr>
<td>2. REC ROR RAT RBCLR</td>
<td>No Always Always No</td>
</tr>
<tr>
<td>3. REC ROR RAT RBCLR</td>
<td>No Always Always Yes</td>
</tr>
</tbody>
</table>
## Hardware

### VMEbus Arbitration

<table>
<thead>
<tr>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Release Every Cycle (REC)</td>
<td>The REC mode causes a release of VMEbus mastership after every VMEbus transfer cycle has been completed. A normal read or write cycle is terminated after the address and data strobes are driven high (inactive state). A read-modify-write cycle (RMW) is terminated after the write cycle is completed by the CPU, through deactivation of the address and data strobes. If the REC mode is enabled, all other bus release functions have no impact (&quot;don’t care&quot;). The REC mode is only for CPU cycles with accesses to the VMEbus and not for cycles initiated by the on-board DMA controller. Programming of the REC mode is described in the <em>FORCE Gate Array FGA-002 User’s Manual</em>.</td>
</tr>
<tr>
<td>Release on Request (ROR)</td>
<td>The ROR mode applies only to CPU cycles to the VMEbus and not for cycles initiated by the FGA-002 DMA controller. In these cases bus mastership is released when another VMEbus board requests bus mastership while the CPU board is the current bus master. For these purposes, the FGA-002 DMA controller can also be the requester causing such a bus release. The ROR mode cannot be disabled, but it is programmable how long the CPU stays VMEbus master in spite of a pending bus request. Programming of the ROR mode is described in the <em>FORCE Gate Array FGA-002 User’s Manual</em>.</td>
</tr>
<tr>
<td>Release After Timeout (RAT)</td>
<td>After every VMEbus access, a 100 µs timer within the FGA-002 starts running. When the timer runs out the CPU board automatically releases its VMEbus mastership. The purpose of the timer is to hold the VMEbus for a short time after every VMEbus transfer, so that the overhead of VMEbus arbitration will be avoided if the CPU makes another VMEbus request within this time period. The timer is only effective for CPU cycles to the VMEbus and not for cycles initiated by the FGA-002 DMA controller. In these cases it is restarted after every VMEbus access, but not before the ROR timer has expired. Therefore, the actual time in which the CPU board holds the bus is approximately equal to the programmed ROR delay time (see above) plus 100 µs. This function cannot be disabled. Programming of the RAT mode is described in the <em>FORCE Gate Array FGA-002 User’s Manual</em>.</td>
</tr>
<tr>
<td>Release on Bus Clear (RBCLR)</td>
<td>The RBCLR mode is only effective for CPU cycles to the VMEbus and not for cycles initiated by the FGA-002 DMA controller. The RBCLR function allows the VMEbus mastership release if an external arbiter asserts the BCLR* signal of the VMEbus. This function then overrides the ROR function timing limitations.</td>
</tr>
</tbody>
</table>
Programming of the RBCLR mode is described in the *FORCE Gate Array FGA-002 User’s Manual*.

**Release When Done (RWD)**
The DMA controller within the FGA-002 can also become VMEbus master. It always operates in transfer bursts (maximum 32 transfers). The bus is always released after completion of such a transfer burst. The other bus release functions are for CPU mastership to the VMEbus only.

**Release on ACFAIL (ACFAIL)**
If the CPU board is programmed to be the ACFAIL handler for the VMEbus system and if the ACFAIL* signal from the VMEbus is asserted, the CPU will not release the VMEbus if it is already the VMEbus master. That is, REC, ROR, RAT, and RBCLR do not operate in this case. If the board is not ACFAIL handler and the ACFAIL* signal is asserted, the board will release the VMEbus immediately.

### 3.16.4 VMEbus Grant Driver

If the CPU board detects itself being plugged in slot 1 (see below), it will automatically use bus grant level 3 (BG3*) and drive the 3 remaining bus grant signals (BG0*, BG1*, and BG2*) to a high level.

### 3.17 VMEbus Slot-1

The SYS68K/CPU-60 may be used as system controller when plugged into slot 1 but the slot-1 functions (see below) are only enabled when the SYS68K/CPU-60 is detected as slot-1 device. The slot-1 functions are also called system controller functions.

**IMPORTANT**

Malfunction

If not on an active backplane,

- remove the jumper on the backplane connecting BG3IN and BG3OUT for the SYS68K/CPU-60 slot.
- assemble the jumpers for BGIN and BGOUT on lower and higher slots on the backplane where no board is plugged.
3.17.1 Slot-1 (System Controller) Functions

When the CPU board is a slot-1 device, the hardware of the SYS68K/CPU-60 sets up the required system controller functions:

- drive SYSCLK to VME (see section 3.17.4 “The SYSCLK Driver” on page 88),
- use VMEbus arbitration level 3, instead of the level selected by SW6-3 and SW6-4 (default “OFF OFF = level 3 (BR3*)”, see page 12),
- drive floating bus grant levels 0, 1, and 2 to a high level signal,
- and allow the SYS68K/CPU-60 bus timer to terminate VME cycles (timeout), if it is enabled (see section 3.17.5 “VMEbus Timer” on page 89).

**IMPORTANT**
The arbiter of the FGA-002 will not automatically be set by hardware when detecting slot-1 by switch setting or auto-detection. It must be enabled by software if the CPU board is system controller (e.g., FGA Boot enables the arbiter automatically). For more information on the FGA-002 arbiter, see the *FORCE Gate Array FGA-002 User’s Manual*.

3.17.2 Slot-1 Detection

Auto-detection The board’s slot-1 auto-detection mechanism probes the VMEbus bus-grant-in-level-3 pin (BG3IN) during power up to see whether it is possible to pull this signal down to a low signal level.

- When the SYS68K/CPU-60 is plugged into slot 1, it will succeed in pulling the VME signal to a low signal level, because BG3IN is floating on slot 1. Hence, the CPU board detects slot 1.
- When the CPU-60 is not plugged into slot 1, it will receive the BG3IN from a board plugged into a lower slot. It will fail trying to pull the VME signal to a low signal level. Hence, the CPU board does not detect slot 1.

Manual detection The following situation may cause the SYS68K/CPU-60 to conclude that slot-1 is detected although being in a different slot:

A VMEbus system begins with the highest daisy-chain priority at slot 1, the left most slot. As the slots move right they lose daisy-chain priority, so slot 2 has higher daisy-chain priority over slot 3, and slot 3 has higher daisy-chain priority over slot 4, and so on. After powering up, auto-detection may fail when another board is plugged into a slot with lower daisy-chain priority. This results in the board (incorrectly) not driving its bus-grant-out-level-3 (BG3OUT) on the VMEbus to the high signal level as defined by the VME specification.
In this situation the SYS68K/CPU-60 probes its BG3IN at a low signal level and concludes that slot 1 is detected. However, the conclusion does not fit the actual system setup. To prevent this mismatch you can

- disable the auto-detection by setting SW6-1 appropriately: ON = disabled (also called manual mode) (default “OFF”, see page 12)
- and enable the slot-1 functions manually by setting SW6-2 appropriately: ON = enabled (default “OFF”, see page 12). For SW6-2 to take any effect SW6-1 must be ON = disabled (also called manual mode).

### 3.17.3 Slot-1 Status Register

The status of the slot-1 detection or manual mode SW6-2 configuration may be read via the slot-1 status register at FF80.1000₁₆. It is a read-only register.

**IMPORTANT**

Malfunction
Writing to the slot-1 status register may cause malfunctions of the CPU board.

- Never write to the slot-1 status register.

#### Table 49 Slot-1 status register (RO)

<table>
<thead>
<tr>
<th>Bit</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>reserved</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>S1STAT</td>
</tr>
</tbody>
</table>

S1STAT indicates whether slot-1 has been detected (by auto-detection or because of switch setting, see "SW6-1" and “SW6-2” on page 12).

= 0 Slot 1 has been detected.

= 1 Slot 1 has not been detected.

### 3.17.4 The SYSCLK Driver

The CPU board contains all necessary circuits to support the SYSCLK signal. The output signal is a stable 16 MHz signal with a 50% duty cycle. The driver circuitry for the SYSCLK signal can source a current of 64 mA.

The SYSCLK signal will be enabled if slot-1 has been detected (by auto-detection or because of switch setting, see "SW6-1" and “SW6-2” on page 12).
3.17.5 VMEbus Timer

The FGA-002 disposes of a bus timer to terminate VME transfers generating a bus error when no acknowledge can be detected after a timeout period.

In addition to the FGA-002 bus timer, the SYS68K/CPU-60 provides a VMEbus timer. This timer can only be enabled when the CPU board provides system controller functions. The SYS68K/CPU-60 VMEbus timer is controlled by the timer within the memory controller. The timeout period can be configured by the register for the timer within the memory controller (see section 3.8.2 “Memory Configuration Register” on page 54).

3.18 Serial I/O – SCC AM 85C30

The 2 serial I/O channels are implemented by using 1 SCC AM 85C30 (serial communication controller, see data sheet “SCC AM 85C30” in section 5). The operating mode and data format of each channel can be programmed independently from each other.

Clock inputs

The peripheral clock inputs of the SCC are driven by a 8-MHz clock. A chip-external on-board quartz provides the 14.7456 MHz clock necessary for baud rates greater than 9600 Baud.

IRQ

The interrupt request of the SCC is connected to the LIRQ5 input of the FGA-002. It is low active. To interrupt acknowledge cycles of the 68060 CPU, the SCC provides its own vector. Instead, the FGA-002 can be programmed to provide the vector.

SCC base address

FF80.200016

Driver modules

To easily vary the serial I/O interfaces according to the application’s needs FORCE COMPUTERS has developed RS-232, RS-422, and RS-485 hybrid modules: the FH-002, FH-003/FH-422T, and FH-007. For each serial I/O channel one of those 21-pin single in-line (SIL) hybrids is installed on-board. The location of the hybrid related to a serial I/O channel is listed in the following table which also shows the serial I/O register map.
For the correct configuration of the serial channels, the connectors which are available and the connectors’ pinout, see section 2.7 “Serial I/O Ports – SCC” on page 17 and section 2.12 “SYS68K/IOBP-1” on page 28.

### 3.18.1 RS-485 Configuration

**FH-007**

For the RS-485 configuration the FORCE COMPUTERS FH-007 hybrid module must be used. It provides 2 enable signals, the RE signal on pin 14 for the receiver and the DE signal on pin 16 for the transmitter, which must be controlled by the serial driver of an operating system.

**RE signal**

The RE signal on pin 14 is connected to the DTR signal of the SCC and can be controlled by bit 7 of the WR5 register as shown in table 51 “Bit 7 of the WR5 register” on page 90. The bit must be cleared to enable the RS-485 receiver. If set to 1, the receiver is disabled. For details how to write an SCC register see data sheet “SCC AM 85C30” in section 5.

**DE signal**

The DE signal on pin 16 of the FH-007 hybrid is connected to the General Purpose I/O (GPIO) port of the SCSI-53C720SE controller. GPIO_0 pin controls the RS-485 transmitter-enable (TX-enable) function for the serial interface channel #1 and GPIO_1 pin controls the RS-485 TX-enable function for the serial interface channel #2. By default, the GPIO pins GPIO_0 and GPIO_1 of the SCSI-53C720SE controller are configured as inputs (powerup default).

### Table 50 Serial I/O channel register map and hybrid locations

<table>
<thead>
<tr>
<th>Address</th>
<th>Serial I/O channel</th>
<th>On-board hybrid location</th>
<th>Register name</th>
</tr>
</thead>
<tbody>
<tr>
<td>FF80.2020₁₆</td>
<td>1</td>
<td>J21</td>
<td>SCC channel A control reg.</td>
</tr>
<tr>
<td>FF80.202₁₆</td>
<td></td>
<td></td>
<td>SCC channel A data reg.</td>
</tr>
<tr>
<td>FF80.200₀₁₆</td>
<td>2</td>
<td>J22</td>
<td>SCC channel B control reg.</td>
</tr>
<tr>
<td>FF80.200₁₁₆</td>
<td></td>
<td></td>
<td>SCC channel B data reg.</td>
</tr>
</tbody>
</table>

### Table 51 Bit 7 of the WR5 register

<table>
<thead>
<tr>
<th>DTR (bit 7)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Receiver enabled</td>
</tr>
<tr>
<td>1</td>
<td>Receiver disabled</td>
</tr>
</tbody>
</table>
To realize the RS-485 interface 2 steps are necessary:

- The corresponding GPIO\(_n\) pin \((n = 0, 1)\) must be configured as output via the General Purpose Control (GPCNTL) register of the SCSI-53C720SE according to table 52 “SCSI-53C720SE GPCNTL register” on page 91.

- The RS-485 interface driver must program the TX-enable function via the General Purpose (GPREG) register of the SCSI-53C720SE according to table 53 “SCSI-53C720SE GPREG register” on page 91.

Table 52  
SCSI-53C720SE GPCNTL register

<table>
<thead>
<tr>
<th>Bit</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>GPIO(_)en1</td>
<td>GPIO(_)en0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Bit 7 - 2  
Do not modify bits 2 through 7.

GPIO\(\_\)en1 controls if GPIO\(_1\) pin is input or output.

- \(= 0\) GPIO\(_1\) is output (RS-485 interface on serial channel #2).
- \(= 1\) GPIO\(_1\) is input (RS-232/RS-422 interface on serial channel #2).

GPIO\(\_\)en0 controls if GPIO\(_0\) pin is input or output.

- \(= 0\) GPIO\(_0\) is output (RS-485 interface on serial channel #1).
- \(= 1\) GPIO\(_0\) is input (RS-232/RS-422 interface on serial channel #1).

Table 53  
SCSI-53C720SE GPREG register

<table>
<thead>
<tr>
<th>Bit</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>GPIO(_)1</td>
<td>GPIO(_)0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Bit 7 - 2  
Do not modify bits 2 through 7.

GPIO\(_1\) controls the RS-485 TX-enable function for serial channel #2.

- \(= 0\) RS-485 transmitter is enabled.
- \(= 1\) RS-485 transmitter is disabled.
GPIO_0 controls the RS-485 TX-enable function for serial channel #1.

- 0 RS-485 transmitter is enabled.
- 1 RS-485 transmitter is disabled.

**NOTICE**
Damage of components
Be sure to set the switches SW12-1 and SW12-3 OFF for the RS-485 configuration before configuring the GPIO ports as outputs.

**IMPORTANT**
The serial interface driver of VMEPROM can be used for RS-232 and RS-422 asynchronous communication only. It does not support the RS-485 configuration.

### 3.19 SCSI – 53C720SE

A Small Computer System Interface (SCSI) controller is built around a 53C720SE (see data sheet “SCSI 53C720SE” in section 5).

**ANSI K3T 9.2 compliant**
The full ANSI K3T 9.2 specification is implemented, supporting all standard SCSI features including arbitration, disconnect, reconnect and parity.

**CAUTION**
As done automatically by FGA Boot the first access to the 53C720SE must set the EA bit in the 53C720SE DCNTL register. Accessing the 53C720SE without the EA bit set will lock the CPU bus.

**IMPORTANT**
- To guarantee correct bus arbitration the fast arbitration mode must be selected by setting the FA bit in the 53C720SE DCNTL register.
- To make SCSI master cycles snoopable the TT1 bit in the 53C720SE CTEST0 register must be set.

**SCRIPTS enhancement**
The 53C720SE based SCSI controller uses its own code fetching and SCSI data transfer from the on-board DRAM. The controller’s processor executes so called SCSI SCRIPTS to control the actions on the SCSI and the CPU bus. Therefore, the controller’s processor is also called SCRIPTS processor. SCSI SCRIPTS is a specially designed language for easy SCSI protocol handling. It substantially reduces the CPU activities. The SCRIPTS processor starts SCSI I/O operations in approximately 500 ns whereas traditional intelligent host adapters require 2…8 ms.
The interrupt request line (IRQ) of the SCSI controller is connected to the LIRQ6 input of the FGA-002. The 53C720SE cannot supply its own vector. Therefore, the local interrupt control register of the FGA-002 has to be programmed to be level-sensitive and to supply the IRQ vector for the SCSI controller.

Single-ended 8-bit SCSI-2 signals are available at row A and C of the VMEbus P2 connector. As a factory option also single-ended 16-bit SCSI-2 signals are available at the VMEbus P2 connector (see section 2.11 “VMEbus P2 Connector Pinout” on page 24). An I/O back panel can be plugged onto the rear side of the backplane to interface the SYS68K/CPU-60 to standard 8-bit or 16-bit SCSI connectors (see section 2.11 “VMEbus P2 Connector Pinout” on page 24).

3.19.1 SCSI Register Map

<table>
<thead>
<tr>
<th>SCSI 53C720SE</th>
<th>base address</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>FFF8.000016</td>
</tr>
</tbody>
</table>

**IMPORTANT**

Unforeseeable interference with the 53C720SE operation

In principal, all 53C720SE registers listed in the 53C720SE data sheet are accessible via the CPU bus. Note, however, that the only register that the 68060 CPU can access while the 53C720SE is executing SCRIPTS is the ISTAT register. Attempts to access other registers will interfere with the operation of the 53C720SE. However, all registers are accessible via SCRIPTS.

- To get the correct address use the information for the big endian bus mode 2 within the 53C720SE data sheet as this is the bus mode the SYS68K/CPU-60 uses.
- While the 53C720SE is executing SCRIPTS
  - access only the ISTAT register.
  - use SCRIPTS to access all other registers.

3.19.2 Communication across the SCSI bus

Communication on the SCSIbus is only allowed between 2 SCSI devices at any given time. There may be a maximum of 8 SCSI devices. Each SCSI device has a SCSI ID assigned.

When 2 SCSI devices communicate on the SCSIbus, one acts as initiator and the target performs the operation. A SCSI device usually has a fixed role as initiator or target, but some devices may be able to assume either role. Certain SCSIbus functions are assigned to the initiator and other functions are assigned to the target:
• The initiator may arbitrate for the SCSI bus and select a particular target. An initiator may address up to seven peripheral devices that are connected to a target. An option allows the addressing of up to 2048 peripheral devices per target using extended messages.

• The target may request the transfer of COMMAND, DATA, STATUS, or other information on the data bus. In some cases, it may arbitrate for the SCSI bus and reselect an initiator for the purpose of continuing an operation.

Transfer modes

Information transfers on the data bus are asynchronous and follow a defined REQ/ACK handshake protocol. One byte of information may be transferred with each handshake. The 53C720SE also supports synchronous operation for the data transfer (see data sheet “SCSI 53C720SE” in section 5).

3.20 Floppy Disk – FDC 37C65C

The CPU board contains a single-chip floppy disk controller, the FDC 37C65C (see data sheet “FDC 37C65C” in section 5). The FDC 37C65C is connected to the DMA controller of the FGA-002.

CAUTION

Damage of components

There are floppy disk drives that provide means to connect the floppy disk drive frame electrically with DC ground, e.g., by inserting a jumper on the floppy disk drive.

• Before installing a floppy disk drive always make sure that the floppy disk drive’s frame is not electrically connected with DC ground.

Floppy disk connectors and pinouts

The installed driver/receiver circuits allow direct connection of 3 1/2" and 5 1/4" floppy disk drives.

An I/O back panel can be plugged onto the rear side of the backplane to interface to mass storage devices (see section 2.11 “VMEbus P2 Connector Pinout” on page 24).

Features of the FDC 37C65C

• Built-in data separator

• Built-in write precompensation

• 128-, 256-, 512-, or 1024-byte sector lengths

• 3 1/2" or 5 1/4" single and double density

• Programmable stepping rate (2 to 6 ms)

• 2 data rate selection options – 16 MHz and 9.6 MHz, controlled via the data rate selection register
• 2 drive selects – DSEL 1 and DSEL 2 – generated by the FDC 37C65C
• 2 signals for motor control – on the SYS68K/CPU-60 they are tied together to build the motor-on signal

**IMPORTANT**

To start the floppy disk data transfer the FGA-002 on-chip DMA controller has to be configured in the following way:

- Set the AUX DMA data direction correctly (see “CIO1 port B data register” on page 42 – F_ADDIR bit).
- Set the FGA-002 AUX DMA according to the FORCE Gate Array FGA-002 User’s Manual.

**FDC connectors and pinout**

The FDC signals are available at the VMEbus P2 connector (see section 2.11 “VMEbus P2 Connector Pinout” on page 24). An I/O back panel can be plugged onto the rear side of the backplane to interface the SYS68K/CPU-60 to standard FDC connectors (see section 2.11 “VMEbus P2 Connector Pinout” on page 24).

**FDC register map**

The registers of the FDC are accessible via the 8-bit local I/O bus (byte mode). The following table shows the register layout of the FDC 37C65C for the SYS68K/CPU-60.

**Table 54 FDC register map**

<table>
<thead>
<tr>
<th>Address</th>
<th>Register name</th>
</tr>
</thead>
<tbody>
<tr>
<td>FF80.3800_16</td>
<td>FDC read main status register (RO)</td>
</tr>
<tr>
<td>FF80.3801_16</td>
<td>Read and write data register</td>
</tr>
<tr>
<td>FF80.3880_16</td>
<td>When read: DCHG register When written: data rate selection register</td>
</tr>
<tr>
<td>FF80.3900_16</td>
<td>Write digital output register (WO)</td>
</tr>
<tr>
<td>FF80.3980_16</td>
<td>Access to toggle EJECT line</td>
</tr>
</tbody>
</table>
3.21 Ethernet – LAN AM 79C965A

The CPU board offers a Local Area Network (LAN) interface based on control logic and the integrated local area communications controller AM 79C965A (see data sheet “LAN AM 79C965A” in section 5, but be aware of the IMPORTANT note concerning the 16-bit mode on page 99).

IEEE-802.3 compliant

The internal Manchester Encoder/Decoder of the AM79C965 is compatible with the IEEE-802.3 specification.

The figure below shows a simplified block diagram of the Ethernet interface.

Figure 9 Block diagram of the Ethernet interface

![Block diagram of the Ethernet interface](image)

**IMPORTANT**

- Set the BSWP bit (byte swap bit) in the AM 79C965A CSR3 register to 1 to ensure correct data transfer (the AM 79C965A is designed for little and big Endian byte ordering).
- Set the INTLEVEL bit in the BCR2 register to 0 to enable correct interrupt generation.

LAN connector and pinout

For the front panel connector and its pinouts see section 2.5 “Front Panel” on page 15 and section 2.10 “Ethernet – LAN” on page 23.

IRQ

The AM 79C965A is able to interrupt the 68060 CPU on a FGA-002 programmable level. It is connected to IRQ #7 of the FGA-002 and must be programmed as level-sensitive and high-active.

Bus error handling

As there is no bus error signalling on the VESA local bus a CPU bus buserror has to be handled in a different way: The AM 79C965A regards a bus error acknowledge as normal acknowledge. Therefore, it does not recognize any failures, e.g., in case of VMEbus transfer errors. However, if a transfer which has been initiated by the AM 79C965A is terminated by a buserror, the VL adaption enters an exception handling which disables busmastership for the AM 79C965A during the next arbitration cycle. If the AM 79C965A requests busmastership during that cycle, it will not get the bus and therefore will generate a timeout and an interrupt and will set the memory error bit within the CSR0 register.
IMPORTANT

The memory error always occurs during the busmastership following the failing busmastership cycle.

Bus error handling (cont.)

After toggeling the LANDEC bit within the BCR register in the RIALTO bus bridge the VL adaption leaves the exception handling.

IMPORTANT

• For proper operation of the AM 79C965A only use the DRAM address space as memory area.

• For correct LAN arbitration set the FGA-002 bus error timeout to 64 µs or less (controlled by VMETIMEOUT in the CTL16 register of the FGA-002).

Ethernet node address

The unique Ethernet node address is permanently stored on-board. It can be displayed by using the VMEPROM INFO command (see section 6.5.9 “INFO – Information about the CPU Board” on page 143). FGA Boot provides a utility function to get the CPU board’s Ethernet address (see section 8.3.5 “Get Ethernet Number” on page 200). The Ethernet address can also be read directly from the ID-ROM via the I²C bus protocol beginning at the ID-ROM offset 32₁₆ (see table 14 “CIO1 port B data register” on page 42).

A unique 48-bit Ethernet address has been assigned to your SYS68K/CPU-60: 00:80:42:0D:xx:xx. The CPU board’s Ethernet address consists of

• a general part indicating the FORCE COMPUTERS CPU board family it is belonging to: 00:80:42:0D for SYS68K/CPU-60,

• followed by the 4-digit CPU board’s serial number: xx:xx. The CPU board’s serial number consists of the last 4 digits of the number printed below the product bar-code on the VMEbus P1 connector. The serial number is always taken from the CPU board which contains the Ethernet logic.

Features of the Ethernet interface

• Compatibility with IEEE 802.3/Ethernet

• Data rate of 10 Mbit per second

• 136-byte transmit and 128-byte receive data buffer between LAN and CPU bus, thus improving overall performance and reducing the risk of network overruns or underruns

• DMA capability

• Interrupt generation
### 3.21.1 Register Access

In order to allow jumperless Ethernet implementations, the AM 79C965A has a software-implemented address relocation mode. The LAN I/O address space is register selectable: see Bridge configuration register (BCR) – “LANDEC (R/W)” on page 50.

#### Initializing

After power up the AM 79C965A will not respond to any access on the CPU bus. However, the AM 79C965A will snoop any I/O write accesses that may be present.

**IMPORTANT**

The AM 79C965A will wait for a sequence of 12 uninterrupted long write accesses to address \(378_{16}\). The 12 long-write accesses must occur without intervening accesses to other locations and they must contain the data in the order shown in the table below.

FGA Boot does this automatically.

**Table 55** Initializing the LAN AM 79C965A register access

<table>
<thead>
<tr>
<th>Access no.</th>
<th>Address</th>
<th>Data [D7…0]</th>
<th>ASCII interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>378(_{16})</td>
<td>41(_{16})</td>
<td>A</td>
</tr>
<tr>
<td>2</td>
<td>378(_{16})</td>
<td>4d(_{16})</td>
<td>M</td>
</tr>
<tr>
<td>3</td>
<td>378(_{16})</td>
<td>44(_{16})</td>
<td>D</td>
</tr>
<tr>
<td>4</td>
<td>378(_{16})</td>
<td>01(_{16})</td>
<td>n/a</td>
</tr>
<tr>
<td>5</td>
<td>378(_{16})</td>
<td>IOBASEL[7:0]</td>
<td>n/a</td>
</tr>
<tr>
<td>6</td>
<td>378(_{16})</td>
<td>IOBASEL[15:8]</td>
<td>n/a</td>
</tr>
<tr>
<td>7</td>
<td>378(_{16})</td>
<td>IOBASEL[23:16]</td>
<td>n/a</td>
</tr>
<tr>
<td>8</td>
<td>378(_{16})</td>
<td>IOBASEL[31:24]</td>
<td>n/a</td>
</tr>
<tr>
<td>9</td>
<td>378(_{16})</td>
<td>BCR2[7:0]</td>
<td>n/a</td>
</tr>
<tr>
<td>10</td>
<td>378(_{16})</td>
<td>BCR2[15:8]</td>
<td>n/a</td>
</tr>
<tr>
<td>11</td>
<td>378(_{16})</td>
<td>BCR21[7:0]</td>
<td>n/a</td>
</tr>
<tr>
<td>12</td>
<td>378(_{16})</td>
<td>BCR21[15:8]</td>
<td>n/a</td>
</tr>
</tbody>
</table>

#### Normal operation

After the CPU board initialization and the Ethernet initialization (see table 57 “Example word-swapped init. block for LAN AM 79C965A in 16-bit mode” on page 100) the AM 79C965A operates without any CPU interaction. It transfers prepared data, receives incoming packets and stores them into reserved memory locations. To signal service requests, the AM 79C965A interrupt signal is connected to the FGA-002’s LIRQ7.
input. The FGA-002 has to be programmed to be level sensitive and to supply the vector, because the AM 79C965A has no provision to do so.

### Register Access

After initialization the AM 79C965A registers are selected by writing the corresponding register number to address \texttt{FFF0.0010}_{16}. Thereafter, the register is accessible at address \texttt{FFF0.0012}_{16}. Both addresses must be accessed with word-size instructions.

### IMPORTANT

The AM 79C965A can be used in two different modes: 16-bit and 32-bit mode.

- Regardless of being in 16-bit or 32-bit mode the Ethernet address is to be initialized in byte-swapped order as documented in table 57 “Example word-swapped init. block for LAN AM 79C965A in 16-bit mode” on page 100.

- **16-bit mode**
  - FGA Boot relocates the AM 79C965A to address \texttt{FFF0.0000}_{16} and leaves it in the 16-bit mode.
  - If you use the AM 79C965A in 16-bit mode, be aware of the fact that in contrast to the statements in the datasheet all registers are word-swapped (see the 2 examples below).

- **32-bit mode**
  - If you use the AM 79C965A in 32-bit mode, remember that no registers are word-swapped.
  - For further information on the 32-bit mode see data sheet “LAN AM 79C965A” in section 5.

### Table 56 Ethernet controller address layout (16-Bit mode)

<table>
<thead>
<tr>
<th>Address</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{FFF0.0010}_{16}</td>
<td>Register address port (RAP)</td>
</tr>
<tr>
<td>\texttt{FFF0.0012}_{16}</td>
<td>Register data port (RDP)</td>
</tr>
<tr>
<td>\texttt{FFF0.0014}_{16}</td>
<td>Bus configuration register data port (BDP)</td>
</tr>
<tr>
<td>\texttt{FFF0.0016}_{16}</td>
<td>Reset register</td>
</tr>
</tbody>
</table>

### Example 1 for word swapping:

The following table shows the initialization block for the AM 79C965A when used in 16-bit mode. It includes the hypothetic Ethernet address \texttt{12:34:56:78:9A:BC} at the addresses \texttt{0}_{16}, \texttt{4}_{16}, and \texttt{6}_{16}. For information on the initialization block and its use see data sheet “LAN AM 79C965A” in section 5.
Example 2 for word swapping:

Initializing the Receive Descriptor: The following table shows the initialization of the receive descriptor in 16-bit mode.

**Table 57**

<table>
<thead>
<tr>
<th>Address</th>
<th>Contents to be written</th>
</tr>
</thead>
<tbody>
<tr>
<td>0_{16}</td>
<td>( \text{eth}<em>1 = 34</em>{16} ) ( \text{eth}<em>0 = 12</em>{16} )</td>
</tr>
<tr>
<td>2_{16}</td>
<td>MODE 15…00</td>
</tr>
<tr>
<td>4_{16}</td>
<td>( \text{eth}<em>5 = BC</em>{16} ) ( \text{eth}<em>4 = 9A</em>{16} )</td>
</tr>
<tr>
<td>6_{16}</td>
<td>( \text{eth}<em>3 = 78</em>{16} ) ( \text{eth}<em>2 = 56</em>{16} )</td>
</tr>
<tr>
<td>8_{16}</td>
<td>LADR 31…16</td>
</tr>
<tr>
<td>A_{16}</td>
<td>LADR 15…00</td>
</tr>
<tr>
<td>C_{16}</td>
<td>LADR 63…48</td>
</tr>
<tr>
<td>E_{16}</td>
<td>LADR 47…32</td>
</tr>
<tr>
<td>10_{16}</td>
<td>RLEN followed by a 0 followed by 3 reserved bits followed by RDRA 23…16</td>
</tr>
<tr>
<td>12_{16}</td>
<td>RDRA 15…0</td>
</tr>
<tr>
<td>14_{16}</td>
<td>TLEN followed by a 0 followed by 3 reserved bits followed by TDRA 23…16</td>
</tr>
<tr>
<td>16_{16}</td>
<td>TDRA 15…0</td>
</tr>
</tbody>
</table>

**Table 58**

<table>
<thead>
<tr>
<th>Address</th>
<th>Contents to be written</th>
</tr>
</thead>
<tbody>
<tr>
<td>0_{16}</td>
<td>Flags BADR 23…16</td>
</tr>
<tr>
<td>2_{16}</td>
<td>BADR 15…8 BADR 7…0</td>
</tr>
<tr>
<td>4_{16}</td>
<td>MCNT 11…8 MCNT 7…0</td>
</tr>
<tr>
<td>6_{16}</td>
<td>BCNT 11…8 BCNT 7…0</td>
</tr>
</tbody>
</table>

**IMPORTANT**

In 16-bit mode the 32-bit address BADR is built from CSR2 (IADR 31…24).
3.22 Reset Generation

The reset affects all on-board modules and chips. When resetting the SYS68K/CPU-60 an automatic self-test routine checks the functional groups of the board. There are 7 sources which may initiate the reset:

1. Voltage sensor unit: The CPU board is reset as long as the supply voltage is below 4.75 V and above 3 V – this is also true during power up. After exceeding the threshold the reset timer will assert the RESET* signal for approximately 200 ms. The reset timer will also be triggered if the voltage has dropped below 4.75V.

2. Front-panel reset key: It triggers the reset timer to generate a reset (see “RESET” on page 15).

3. Watchdog timer: If the reset is generated by the watchdog timer, the WDIRQ bit in the MDR is set to 1 (see section 3.5 “Watchdog Timer” on page 48). The watchdog reset is a pseudo power up.

4. RESETOUT bit in the memory configuration register: If the RESETOUT bit in the memory configuration register is set to 1, a reset is generated (see table 26 “MCR, memory configuration register” on page 54). Setting the RESETOUT bit has the same effect as a reset generated by the watchdog timer.

5. 68060 CPU reset call: A 68060 CPU reset call is triggered, when the 68060 CPU addresses the FGA-002 at FFD0.0E00₁₆ – either in a read or in a write cycle. The 68060 CPU reset call has the same effect as toggling the reset key.

6. 68060 CPU reset instruction: The 68060 CPU reset instruction is designed to reset peripherals under program control, without resetting the 68060 CPU itself. This instruction is fully supported by the CPU board. The reset instruction triggers the reset generator and resets all on-board modules and chips driving reset to low. The external logic enters boot mode. Therefore, the shared memory at location 0000.0000₁₆ will be disabled causing a failure of the program executed from the shared RAM. To run a reset instruction correctly, the reset instruction has to be executed from a local bus memory, for example, the local SRAM. Also the execution from the system PROM is possible, but be sure that no DRAM access is necessary (e.g. no stack operation). After the reset instruction one of the RIALTO bus bridge registers has to be accessed to re-enable the shared memory.

If VMEbus SYSRESET* input is asserted before the reset generated by a reset instruction is finished, the processor will still not be reset because of lockout logic.
Additionally, SYSRESET* output is asserted by the FGA-002 if it is enabled via the FGA-002 CTRL9 register.

VMEbus SYSRESET

7. VMEbus SYSRESET: The VMEbus SYSRESET line is received by the SYS68K/CPU-60 only if SYSRESET input is enabled that is if SW9-4 is set appropriately: OFF = enabled.

SYSRESET generation

Whenever a reset is generated by one of the sources 1 to 6, SYSRESET output is asserted additionally if SW9-3 is set appropriately: OFF = enabled (default “OFF”, see page 13). SYSRESET output is asserted by the IEEE 1014 compatible SYSRESET* driver installed on the CPU board. The reset generation circuitry operates when the power supply voltage Vcc reaches approximately 3 volts. An asserted SYSRESET output signal will be held low (active) for at least 200 ms after all conditions that caused the SYSRESET assertion have been removed.

IMPORTANT

The VME SYSRESET generation must be enabled by SW9-3 if the SYS68K/CPU-60 is installed in slot 1 (see section “SYSRESET* input” on page 76).

Reset period

During power up or after activation of the front-panel reset key the CPU board is reset for approximately 200 ms.

Status information

The front-panel RUN LED shows the status of the RESET line (see “RUN” on page 16). If RESET is active, the LED is illuminated red. The LED turns to green if reset is inactive and the processor is not in the halt state.

Initial Supervisor Stack Pointer and Program Counter after Reset

The first 2 read cycles after reset of the 68060 CPU are operand fetches of the initial supervisor stack pointer (ISP) and the initial program counter (IPC). These operands are always fetched from addresses 0000.000016 and 0000.000416, respectively.

IMPORTANT

After reset, the boot PROM is mapped to address 0000.000016. After initialization the firmware enables the DRAM at 0000.000016 with an access to any of the 2 RIALTO registers (see section 3.8.8 “DRAM Access from the 68060 CPU” on page 59).
### 3.23 Information on Front Panel Devices

<table>
<thead>
<tr>
<th>Device</th>
<th>Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>RESET key</td>
<td>See section 3.22 “Reset Generation” on page 101 and section “RESET” on page 15</td>
</tr>
<tr>
<td>ABORT key</td>
<td>see “ABORT” on page 15</td>
</tr>
<tr>
<td>7-Segment hexadecimal display – DIAG</td>
<td>The status display register is located in CIO2 (see section 3.3.5 “Board ID and DIAG Display” on page 44).</td>
</tr>
<tr>
<td>Rotary switches</td>
<td>For information on the status register of the rotary switches see section 3.3.3 “MODE x Rotary Switch Setting” on page 43.</td>
</tr>
<tr>
<td>SYSF LED</td>
<td>See section 3.15.1 “Exception Signals SYSFAIL, SYSRESET, and AC-FAIL” on page 76.</td>
</tr>
<tr>
<td>UL LED</td>
<td>See section 3.6.2 “Bridge Configuration Register” on page 50.</td>
</tr>
</tbody>
</table>
Please Note…

The circuit schematics section is an integral part of the SYS68K/CPU-60 Technical Reference Manual (P/N 204077). Yet, it is packaged separately to enable easy updating.

The circuit schematics section will always be shipped together with the Technical Reference Manual.

Please:

☞ Insert the circuit schematics section (P/N 204075) now into the SYS68K/CPU-60 Technical Reference Manual (P/N 204077).

☞ Remove this sheet.
4  Circuit Schematics

Copies of the SYS68K/CPU-60 are found on the following pages.
Please Note…

The data sheet section is an integral part of the *SYS68K/CPU-60 Technical Reference Manual* (P/N 204077). Yet, it is packaged separately to enable easy updating.

The data sheet section will always be shipped together with the *Technical Reference Manual*.

Please:

☞ **Insert the data sheet section (P/N 204076) now into the SYS68K/CPU-60 Technical Reference Manual (P/N 204077).**

☞ **Remove this sheet.**
5 Data Sheets

This is a list of all data sheets that are relevant for the SYS68K/CPU-60. Copies of these data sheets are found on the following pages.

1. CIO Z8536
2. FDC 37C65C
3. LAN AM 79C965A
4. RTC 72421
5. SCC AM 85C30
6. SCSI 53C720SE
5.1 CIO Z8536
5.2 FDC 37C65C
5.3 LAN AM 79C965A

This data sheet copy includes

- Application note: PCnet Family Software Design Considerations
5.4 RTC 72421
5.5 SCC AM 85C30
5.6 SCSI 53C720SE

This data sheet copy intentionally only includes the following chapters which contain all information relevant for the SYS68K/CPU-60:

- Purpose and Audience
- Additional Information
- Contents (not stripped to the information included in this copy)
- Chapter 1: Introduction
- Chapter 2: Functional Description
- Chapter 5: Registers
- Chapter 6: Instruction Set of the I/O Processor
- Appendix A: Register Summary
This CPU board operates under the control of VMEPROM, a ROM resident real-time multiuser multitasking monitor program. VMEPROM provides the user with a debugging tool for single and multitasking real-time applications.

All common commands and system calls are described in the *VMEPROM Version 2/32 User’s Manual*.

This section describes those parts of VMEPROM which pertain to the hardware of this CPU board.

- Configuration of the board
- Line assembler/disassembler
- Numerous commands for program debugging, including breakpoints, tracing, processor register display and modify
- Display and modify floating point data registers
- S-record up- and downloading from any port defined in the system
- Time stamping of user programs
- Built-in benchmarks
- Support of RAM-disk and Winchester disks, also allowing disk formatting and initialization
- Disk support for ISCSI-1 cards
- Serial I/O support for up to two SIO-1/2 or ISIO-1/2 boards
- EPROM programming utility using the SYS68K/RR-2/3 boards
- On-board flash memory (boot PROM, system PROM and user flash) programming utility
- Full-screen editor
- Numerous commands to control the PDOS kernel and file manager
- Complete task management
- I/O redirection to files or ports from the command line
- Shell with over 80 commands
- Over 100 system calls to the kernel supported
- Data conversion and file management functions
- Task management system calls in addition to terminal I/O functions
- Starting an application
### 6.1 Power-up Sequence

The power-up sequence is executed upon power up or after resetting the SYS68K/CPU-60. All steps documented below also apply for the reset case unless explicitly stated otherwise.

<table>
<thead>
<tr>
<th>Power-up sequence configuration</th>
<th>The 2 front-panel rotary switches of the CPU board define the actions taken by VMEPROM after power up or reset (see section 6.2.3 “Rotary Switches” on page 128).</th>
</tr>
</thead>
<tbody>
<tr>
<td>FGA Boot</td>
<td>After power up the processor retrieves the initial stack pointer and program counter from address locations 0, 16 and 4, 16. These locations are the first 8 bytes of the Boot ROM area where the FGA boot software (called FGA Boot) resides. They are mapped down to address 0, 16 for a defined start. Afterwards the boot software is executed (see section 8.1 “Boot Sequence” on page 179).</td>
</tr>
<tr>
<td>BIOS modules of VMEPROM</td>
<td>After the boot software has been executed, control is transferred to the BIOS modules of VMEPROM to perform all the necessary hardware initialization of the CPU. The real-time kernel is started and the user interface of VMEPROM is invoked as the first task. The real-time clock (RTC) of the CPU board is read and the software clock of the kernel initialized.</td>
</tr>
</tbody>
</table>
| Terminal connection             | If a terminal is connected to the front-panel serial I/O port 1, the power-up sequence will be terminated by displaying the following 2 messages  

- the VMEPROM banner  
- and the VMEPROM prompt:  

?  

VMEPROM is then ready to accept commands. |
| In case of no messages          | If the above messages do not appear, check the following:  

1. Check the terminal for the setting of the baud rate and the character format. For the default port setup, see section 2.7 “Serial I/O Ports – SCC” on page 17.  

2. Check the cable connection between the CPU board and the terminal. For the serial I/O port pinout and its default setup, see section 2.7 “Serial I/O Ports – SCC” on page 17.  

3. Check the power supply for the presence of +5 V, +12 V, –12 V. For the power consumption of the CPU board, see table 1 “Specification for the SYS68K/CPU-60 board” on page 2.
6.2 Front Panel Related VMEPROM Features

6.2.1 Reset Key

Activating the reset key on the front panel causes

• all programs to terminate immediately
• and the processor and all I/O devices to be reset.

**IMPORTANT**

Loss of data and user program

When the VMEPROM kernel is started, it overwrites the first word in the user memory after the task control block with an exit system call (XEXT). If breakpoints are defined and a user program is running when the reset key is activated, the user program will possibly be destroyed.

• While a program is running only activate the reset key as a last resort when all other actions (such as pressing `c twice or aborting the program) have failed.

6.2.2 Abort Key

VMEPROM causes a level-7 interrupt when the abort key is activated. This interrupt cannot be disabled and is therefore the appropriate way to terminate a user program and return to the VMEPROM command level.

**IMPORTANT**

User program tasks with port 0 (phantom port) as their input port will not be terminated.

Activating the abort key while a user program is running causes

• all user registers to be saved at the current location of the program counter
• and the message Aborted Task to be displayed along with the contents of the processor register.

Activating the abort key while a VMEPROM built-in command is executed or the command interpreter is waiting for input causes

• the message Aborted Task to be displayed (contrary to the situation above, the processor registers are neither modified nor displayed)
• and the control to be transferred to the command interpreter.
6.2.3 Rotary Switches

The settings of the 2 rotary switches on the front panel of the CPU board are read in by VMEPROM after power up or reset. They define the actions taken by VMEPROM after power up or reset:

- rotary switch MODE 1 controls
  - the program invoked,
  - the start-up file executed,
  - and the check of the VMEbus for available hardware.

- rotary switch MODE 2 controls
  - the initialization of the RAM disk,
  - the default data size on the VMEbus,
  - and the memory location of the RAM disk.

All settings documented below apply for both, during power-up and during reset unless explicitly stated otherwise.

Configuring the action-to-switch-setting correspondence

The correspondence between the actions VMEPROM takes after power up and reset and the rotary switch settings can be configured by patching the system PROM according to the user’s choices. For a description of the memory locations to be patched, see section 7.7 “Modifying Special Locations in ROM” on page 174.

Default correspondence

The following tables show frequently used configuration examples and document the default correspondence between actions and the rotary switch settings as defined for VMEPROM. For the correspondence between the rotary switch setting and the bits set in the MODE x status register, see section 3.3.3 “MODE x Rotary Switch Setting” on page 43.
Table 59 Examples for power-up and reset configuration by rotary switches

<table>
<thead>
<tr>
<th>Setting of MODE</th>
<th>Description of actions after power-up and reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 1</td>
<td></td>
</tr>
<tr>
<td>FF</td>
<td>• 32-bit VMEbus data size</td>
</tr>
<tr>
<td></td>
<td>• RAM disk at top of memory</td>
</tr>
<tr>
<td></td>
<td>• Start of VMEPROM</td>
</tr>
<tr>
<td>4 C</td>
<td>• RAM disk initialization</td>
</tr>
<tr>
<td></td>
<td>• 32-bit VMEbus data size</td>
</tr>
<tr>
<td></td>
<td>• RAM disk at address $4080\cdot0000_{16}$</td>
</tr>
<tr>
<td></td>
<td>• Start of VMEPROM</td>
</tr>
<tr>
<td></td>
<td>• Execution of start-up file SY$STRT$</td>
</tr>
<tr>
<td></td>
<td>• Check for available hardware on the VMEbus and wait for SYSFAIL to disappear from the VMEbus</td>
</tr>
<tr>
<td>B 3</td>
<td>• 16-bit VMEbus data size</td>
</tr>
<tr>
<td></td>
<td>• Start of user program at $4080\cdot0000_{16}$</td>
</tr>
<tr>
<td>MODE 2 status register bit</td>
<td>Description and MODE 2 setting at front panel</td>
</tr>
<tr>
<td>---------------------------</td>
<td>------------------------------------------------</td>
</tr>
</tbody>
</table>
| 3 | Bit 3 defines whether the RAM disk will be initialized after power-up and reset.  
   Bit 3 = 0 (settings 0 through 7)  
   The RAM disk is initialized after power-up and reset as defined by bit 0 and bit 1. After disk initialization all data on the disk is lost.  
   Bit 3 = 1 (settings 8 through F)  
   The RAM disk will not be initialized after power up and reset. |
| 2 | Bit 2 defines the default data size on the VMEbus.  
   Bit 2 = 0 (settings 0 – 3, 8 – B)  
   The default data size is 16 bit.  
   Bit 2 = 1 (settings 4 – 7, C – F)  
   The default data size is 32 bit. |
| 1 and 0 | Bit 1 and bit 0 define the default RAM disk usage.  
   Bit 1 = 0 and bit 2 = 0 (settings 0, 4, 8, C)  
   RAM disk at 4080.0000₁₆ (512 Kbyte)  
   Bit 1 = 0 and bit 2 = 1 (settings 1, 5, 9, D)  
   RAM disk at FF0.8000₁₆ (64 Kbyte)  
   Bit 1 = 1 and bit 2 = 0 (settings 2, 6, A, E)  
   RAM disk at FC80.0000₁₆ (512 Kbyte)  
   Bit 1 = 1 and bit 2 = 1 (settings 3, 7, B, F)  
   RAM disk at top of memory (32 Kbyte) |
### Table 61: Power-up and reset actions defined by rotary switch MODE 1

<table>
<thead>
<tr>
<th>MODE 1 status register bit</th>
<th>Description and MODE 1 setting at front panel</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 and 2</td>
<td>Bit 3 and bit 2 define the program to be invoked after power-up and reset.</td>
</tr>
<tr>
<td></td>
<td>Bit 3 = 0 and bit 2 = 0 (settings 0, 1, 2, 3)</td>
</tr>
<tr>
<td></td>
<td>The user program at 4080.0000₁₆ is invoked.</td>
</tr>
<tr>
<td></td>
<td>Bit 3 = 0 and bit 2 = 1 (settings 4, 5, 6, 7)</td>
</tr>
<tr>
<td></td>
<td>The user program at FFC0.8000₁₆ is invoked.</td>
</tr>
<tr>
<td></td>
<td>Bit 3 = 1 and bit 2 = 0 (settings 8, 9, A, B)</td>
</tr>
<tr>
<td></td>
<td>The user program at FC80.0000₁₆ is invoked.</td>
</tr>
<tr>
<td></td>
<td>Bit 3 = 1 and bit 2 = 1 (settings C, D, E, F)</td>
</tr>
<tr>
<td></td>
<td>VMEPROM is invoked.</td>
</tr>
<tr>
<td>1</td>
<td>Bit 1 defines whether VMEPROM tries to execute a start-up file after power-up and reset.</td>
</tr>
<tr>
<td></td>
<td>Bit 1 = 0 (settings 0, 1, 4, 5, 8, 9, C, D)</td>
</tr>
<tr>
<td></td>
<td>VMEPROM tries to execute a start-up file. The default filename is SY$STRT.</td>
</tr>
<tr>
<td></td>
<td>Bit 1 = 1 (settings 2, 3, 6, 7, A, B, E, F)</td>
</tr>
<tr>
<td></td>
<td>VMEPROM does not try to execute a start-up file but comes up with the default banner, instead.</td>
</tr>
<tr>
<td>0</td>
<td>Bit 0 defines whether VMEPROM takes the following two actions:</td>
</tr>
<tr>
<td></td>
<td>• check the VMEbus for availability of any of the following hardware:</td>
</tr>
<tr>
<td></td>
<td>Contiguous memory, ISIO-1/2, SIO-1/2, ISCSI-1, WFC-1</td>
</tr>
<tr>
<td></td>
<td>• wait for SYSFAIL to disappear from the VMEbus.</td>
</tr>
<tr>
<td></td>
<td>For details, see section 6.5.2 “CONFIG – Search VMEbus for Hardware” on page 136.</td>
</tr>
<tr>
<td></td>
<td>Bit 0 = 0 (settings 0, 2, 4, 6, 8, A, C, E)</td>
</tr>
<tr>
<td></td>
<td>VMEPROM takes both actions.</td>
</tr>
<tr>
<td></td>
<td>Bit 0 = 1 (settings 1, 3, 5, 7, 9, B, D, F)</td>
</tr>
<tr>
<td></td>
<td>VMEPROM does not take any of the 2 actions.</td>
</tr>
</tbody>
</table>
## 6.3 Memory Usage of VMEPROM

### 6.3.1 Default Memory Usage of VMEPROM

By default, VMEPROM uses the following memory assignment for the CPU board:

<table>
<thead>
<tr>
<th>Start address</th>
<th>End address</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000.000016</td>
<td>0000.03FF16</td>
<td>Vector table</td>
</tr>
<tr>
<td>0000.040016</td>
<td>0000.0FFF16</td>
<td>System configuration data</td>
</tr>
<tr>
<td>0000.100016</td>
<td>0000.5FFF16</td>
<td>SYRAM</td>
</tr>
<tr>
<td>0000.600016</td>
<td>0000.6FFF16</td>
<td>VMEPROM internal use</td>
</tr>
<tr>
<td>0000.700016</td>
<td>0000.7FFF16</td>
<td>Task control block 0</td>
</tr>
<tr>
<td>0000.800016</td>
<td>........</td>
<td>User memory of task 0</td>
</tr>
<tr>
<td>........</td>
<td>........</td>
<td>Mail array</td>
</tr>
<tr>
<td>........</td>
<td>........</td>
<td>RAM disk (optional)</td>
</tr>
<tr>
<td>........</td>
<td>End of local memory</td>
<td>Hashing buffers for disk I/O</td>
</tr>
</tbody>
</table>

**IMPORTANT**

The size of the first task cannot be extended beyond the highest on-board memory address. If more memory is available (on VMEbus), it can only be used for data storage, but not for tasking memory.

### 6.3.2 Default ROM Use of VMEPROM

The following table shows the use of the system flash memory including VMEPROM. Note that only the first 512 Kbyte will be used by VMEPROM, the remaining space is available for user applications.

For detailed information about user alterable locations see section 7.7 “Modifying Special Locations in ROM” on page 174 and section 7.8 “Binding Applications to VMEPROM” on page 177.
### 6.4 Devices and Interrupts Used by VMEPROM

#### 6.4.1 Addresses of the On-Board I/O Devices

The following table shows the on-board I/O devices and their addresses:

<table>
<thead>
<tr>
<th>Base address</th>
<th>Device</th>
</tr>
</thead>
<tbody>
<tr>
<td>FF80.0C00_{16}</td>
<td>CIO1 Z8536</td>
</tr>
<tr>
<td>FF80.0E00_{16}</td>
<td>CIO2 Z8536</td>
</tr>
</tbody>
</table>
### 6.4.2 On-Board Interrupt Sources

The following table shows the on-board interrupt sources and levels defined by VMEPROM. All interrupt levels and vectors of the on-board I/O devices are software programmable via the FGA-002 gate array.

<table>
<thead>
<tr>
<th>Device</th>
<th>IRQ level</th>
<th>Vector number</th>
<th>Vector address</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abort switch</td>
<td>7</td>
<td>232 E8&lt;sub&gt;16&lt;/sub&gt;</td>
<td>3A0&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>FGA-002 DMA error</td>
<td>4</td>
<td>235 EB&lt;sub&gt;16&lt;/sub&gt;</td>
<td>3AC&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>FGA-002 DMA ready</td>
<td>4</td>
<td>236 EC&lt;sub&gt;16&lt;/sub&gt;</td>
<td>3B0&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>Watchdog</td>
<td>7</td>
<td>240 F0&lt;sub&gt;16&lt;/sub&gt;</td>
<td>3C0&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>CIO1 (timer tic)</td>
<td>5</td>
<td>242 F2&lt;sub&gt;16&lt;/sub&gt;</td>
<td>3C8&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>SCC</td>
<td>4</td>
<td>244 F4&lt;sub&gt;16&lt;/sub&gt;</td>
<td>3D0&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
</tbody>
</table>

### 6.4.3 Off-Board Interrupt Sources

VMEPROM supports several VMEbus boards. As these boards are interrupt driven, the level and vectors must be defined for VMEPROM to work properly. The following table shows the default setup of the interrupt levels and vectors of the supported hardware.

For a detailed description of the boards’ hardware setup, see section 7 “Appendix to VMEPROM” on page 151.

For further information on the supported I/O boards together with the base addresses and the interrupt levels and vectors, see table below. In order to ensure that these boards work correctly with VMEPROM, the listed interrupt vectors must not be used.
6.4.4 The On-Board Real-Time Clock

During the power-up sequence the on-board real-time clock of the CPU board is read and the current time is loaded into VMEPROM. This sequence is done automatically and requires no user intervention. If the software clock of VMEPROM is set by the \texttt{ID} command, the RTC is automatically set to the new time and date values.

6.5 VMEPROM Commands

The VMEPROM commands are resident and available at any time.

- **Common commands**: Most of the commands are common for all versions of VMEPROM. For a description of all common VMEPROM commands and for an in depth description of VMEPROM itself refer to the \textit{VMEPROM Version 2/32 User's Manual}.

- **CPU board commands**: VMEPROM commands which are specific for the hardware of the CPU board are described in this section.

- **Quick overview**: The \texttt{HELP} command provides a short description of all available VMEPROM commands.
  - Enter \texttt{HELP} for a description of all commands.
  - Enter \texttt{HELP <command>} for a description of the command \texttt{command}.

- **Command line syntax**: All VMEPROM commands use the following format:

  \texttt{? <command> parameters}
In some cases commands do not use parameters, at all. If 2 or more parameters are entered, they must be separated by a space or a comma.

### 6.5.1 ARB – Set the Arbiter of the CPU Board

**Format**

ARB

The `ARB` command allows the user to set the arbitration modes and the release modes of the CPU board for the VMEbus. Additionally, the VMEbus interrupts can be enabled or disabled.

**Example:**

```plaintext
? ARB
Set arbiter mode for VME-BUS:
STATUS : ROR & RAT & RBCLR & FAIR
SET : Release on bus clear (RBCLR) (Y/N) ? \(Y\)
SET : Fair VME-BUS arbitration (FAIR) (Y/N) ? \(N\)

Enable(1) / Disable(0) VMEbus interrupts by level:
STATUS : Level: 7 6 5 4 3 2 1
         \(1111111\)
SET : Enter new interrupt mask: \(1111110\)

? _
```

### 6.5.2 CONFIG – Search VMEbus for Hardware

**Format**

CONFIG

This command searches the VMEbus for available hardware regardless of the rotary switch setting and enables installation of additional memory. The `CONFIG` command also installs Winchester disks in the system and initializes the disk controller (if available).

If a SYSFAIL is active on the VMEbus (e.g. being generated by an ISIO-1/2 or ISCSI-1 controller during self-test), the command is suspended until the SYSFAIL signal is no longer active.

**IMPORTANT**

Additional memory installation

- All boards to be installed must use the addresses documented in section 7.1 “Driver Installation” on page 151.
- Install additional memory only by using this command.
Automatic
memory
detection

Additional memory must be contiguous to the on-board memory of the CPU board. This memory is cleared by the `CONFIG` command to allow DRAM boards with parity to be used. Please remember that the installation of additional memory does not affect the RAM size of the running task. However, VMEPROM identifies the installed memory area and every time memory is required (e.g., by CT or FM) it is taken from this area as long as there is enough free space.

The following hardware is detected when issuing the command:

1. ISIO-1/2
2. SIO-1/2
3. ISCSI-1
4. WFC-1
5. Contiguous memory starting at the highest on-board memory address

For the setup of all supported boards, see section 7 “Appendix to VMEPROM” on page 151.

Example:

```
? CONFIG
UART FORCE ISIO-1/2 (U3) INSTALLED
ISIO-1/2: 1 boards available
?
```

### 6.5.3 FERASE – Erase Flash Memories

**Format**

```
FERASE flashbank
FERASE flashbank, flashoffset, length
```

The `FERASE` command allows erasing flash memory banks.

- Format 1 of the command erases the whole flash memory bank.
- Format 2 allows specifying a region to erase.

**IMPORTANT**

This region must exactly match the page boundaries of the flash devices.

**Example:**

If the SYS_FLASH bank consists of four 28F008 (1 M * 8 bit) devices in parallel with a page size of 64 Kbyte each, the minimum size of one erasable region is 256 Kbyte (64 KB * 4 devices).
Parameters

\textit{\texttt{flashbank}}

Symbolic name or base address of the flash memory bank that should be erased. The following symbolic names are supported:

- \texttt{BOOT\_FLASH} (first) boot flash
- \texttt{BOOT\_FLASH1} first boot flash
- \texttt{BOOT\_FLASH2} second boot flash
- \texttt{SYS\_FLASH} system flash
- \texttt{USER\_FLASH} user flash

\textit{\texttt{flashoffset}}

Optional relative byte offset within the flash bank.

\textit{\texttt{length}}

Optional length in bytes. If \texttt{flashoffset} and \texttt{length} are not specified, the whole bank will be erased.

\underline{Example:}

\begin{verbatim}
? FERASE
Usage: FERASE <flashbank>,[<flashoffset>,<length>]
Parameter <flashbank> is the base address of the flash bank
or one of the following defines:
    BOOT\_FLASH1 BOOT\_FLASH2 SYS\_FLASH1 USER\_FLASH1

? FERASE BOOT\_FLASH2
Erasing flash memory ... done.

? _
\end{verbatim}

6.5.4 \textbf{FGA – Change Boot Setup for Gate Array}

\textbf{Format} \quad \texttt{FGA}

Some registers of the gate array can be defined by the user. The contents of these registers are stored in the on-board battery-buffered SRAM in a short form.

The boot software for the gate array will take these values after reset to initialize the gate array. The \texttt{FGA} command may be used to enter an interactive node for changing this boot table in the battery-buffered SRAM.

The \texttt{FGA} command will show the actual value stored in the battery-buffered SRAM. To change any value, a new one has to be entered in binary format. If only a \texttt{<CR>} is entered, no change will be made. To step backwards a minus has to be entered. If a \texttt{<.>} or \texttt{<ESC>} is given, the \texttt{FGA} command returns to the shell.
The command uses cursor positioning codes of the selected terminal. Use the `st` command to set the correct terminal.

Example:

```
? FGA
>>> Setup for FGA-002 BOOTER <<<

<table>
<thead>
<tr>
<th>Register</th>
<th>FGA offset</th>
<th>value in SRAM</th>
<th>changed value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPECIAL</td>
<td>$0420</td>
<td>$00100000</td>
<td>$00100000</td>
</tr>
<tr>
<td>CTL_01</td>
<td>$0238</td>
<td>$00000111</td>
<td>$00000111</td>
</tr>
<tr>
<td>CTL_02</td>
<td>$023C</td>
<td>$00001011</td>
<td>$00001011</td>
</tr>
<tr>
<td>CTL_05</td>
<td>$0264</td>
<td>$00001100</td>
<td>$00001100</td>
</tr>
<tr>
<td>CTL_12</td>
<td>$032C</td>
<td>$00110011</td>
<td>$00110011</td>
</tr>
<tr>
<td>CTL_14</td>
<td>$0354</td>
<td>$01111110</td>
<td>$01111110</td>
</tr>
<tr>
<td>CTL_15</td>
<td>$0358</td>
<td>$01000000</td>
<td>$01000000</td>
</tr>
<tr>
<td>CTL_16</td>
<td>$035C</td>
<td>$00100000</td>
<td>$00100000</td>
</tr>
<tr>
<td>MBX_00</td>
<td>$0000</td>
<td>$00000000</td>
<td>$00000000</td>
</tr>
<tr>
<td>MBX_01</td>
<td>$0004</td>
<td>$00000000</td>
<td>$00000000</td>
</tr>
<tr>
<td>MBX_02</td>
<td>$0008</td>
<td>$00000000</td>
<td>$00000000</td>
</tr>
<tr>
<td>MBX_03</td>
<td>$000C</td>
<td>$00000000</td>
<td>$00000000</td>
</tr>
<tr>
<td>MBX_04</td>
<td>$0010</td>
<td>$00000000</td>
<td>$00000000</td>
</tr>
<tr>
<td>MBX_05</td>
<td>$0014</td>
<td>$00000000</td>
<td>$00000000</td>
</tr>
<tr>
<td>MBX_06</td>
<td>$0018</td>
<td>$00000000</td>
<td>$00000000</td>
</tr>
<tr>
<td>MBX_07</td>
<td>$001C</td>
<td>$00000000</td>
<td>$00000000</td>
</tr>
</tbody>
</table>
```

6.5.5 FLUSH – Set Buffered Write Mode

Format

`FLUSH`  
`FLUSH ?`  
`FLUSH ON`  
`FLUSH OFF`

This command

- flushes all modified hashing buffers for disk writing or
- enables/disables buffered write mode for the local SCSI controller.

If no argument is entered, all modified hashing buffers are flushed. If the argument `ON` or `OFF` is given, the buffered write mode will be enabled or disabled. When entering a question mark, only a message will be displayed which indicates whether the buffered write mode is enabled or disabled.
6.5.6 FMB – FORCE Message Broadcast

Format

```
FMB slotlist,FMB channel,message
FMB [FMB channel]
```

The *FMB* command allows
- sending a byte message to individual slots in the backplane,
- broadcasting to all boards, and
- getting a pending message.

Format 1 is used to send a message.

Parameters

- **slotlist** is a list of slot numbers and is used to select the slots to which a message is sent. Slot numbers are separated by a ‘/’ sign; a ‘-’ between two slot numbers defines a range of slot numbers. Slot numbers can range from 0 to 21. 0 causes the message to be sent to all slots.

- **FMB channel** defines which FMB channel is used. It can be 0 or 1.

- **message** is the byte message to be deposited into the FMB channel(s).

Format 2 is used to get messages. If no parameter is given, one message of each FMB channel is fetched and displayed. If **FMB channel** is specified, only this channel is addressed and the message will be displayed.

For detailed information on the FORCE message broadcast, see the *FORCE Gate Array FGA-002 User’s Manual*. 

Example:

```
? FLUSH
All modified buffers are flushed
? FLUSH ON
Buffered write is enabled
```
Example:

```plaintext
? FMB
FMB channel 0 is empty
FMB channel 1 is empty

? FMB 1-21,0,$EF

? FMB 1-21,1,%10100001

? FMB
FMB channel 0 = $EF
FMB channel 1 = $A1
? FMB 1-21,1,$77

? FMB
FMB channel 0 is empty
FMB channel 1 = $77

? FMB 1/2/5/7-19/21,0,$1

? FMB
FMB channel 0 = $01
FMB channel 1 is empty

? _
```

### 6.5.7 FPROG – Program Flash Memories

**Format**

- `FPROG flashbank, source`
- `FPROG flashbank, source, flashoffset`
- `FPROG flashbank, source, flashoffset, length`

The `FPROG` command allows programming flash memory banks.

Format 1 of the command programs the whole flash memory bank with the data stored at the specified source address.

Format 2 additionally allows specifying a destination offset within the flash memory bank and programs all the remaining space (from offset to end of flash bank).

Format 3 of the command also specifies the number of bytes to program.

**IMPORTANT**

If the flash memory is not empty, it must be erased before reprogramming it (see section 6.5.3 “FERASE – Erase Flash Memories” on page 137).
Parameters

`flashbank`
Symbolic name or base address of the flash memory bank that should be programmed. The following symbolic names are currently supported:

- `BOOT_FLASH` (first) boot flash
- `BOOT_FLASH1` first boot flash
- `BOOT_FLASH2` second boot flash
- `SYS_FLASH` system flash
- `USER_FLASH` user flash

`source`
Source address of the data to program.

`flashoffset`
Optional relative byte offset within the flash bank. If no offset is specified, 0 is assumed.

`length`
Optional length in bytes. If no length is specified, all the remaining space of the flash bank will be programmed.

Example:

Partly programming the second Boot Flash

```
? FPROG BOOT_FLASH2,100000,0,1375
Programming flash memory                                           0 |###########################################| 100%
Done.
```

6.5.8 FUNCTIONAL – Perform Functional Test

Format

FUNCTIONAL

IMPORTANT

This command is designed for FORCE COMPUTERS internal purposes only.
6.5.9 INFO – Information about the CPU Board

Format

INFO
INFO VME

The first format is used to display information about the CPU board as documented in the VMEPROM Version 2/32 User's Manual. Additionally, the Ethernet address is displayed.

The second format displays the current setting of the VMEbus A32 and A24 slave window.

6.5.10 MEM – Set Data Bus Width of the VMEbus

Format

MEM
MEM 16
MEM 32

This command displays or sets the data bus width of the CPU board on the VMEbus.

- To display the current data bus width enter MEM without arguments.
- To set the data bus width to 16 bits or 32 bits enter MEM 16 or MEM 32, respectively.

If the data bus width is set to 16 bit, long accesses (32-bit) will be translated into 2 word accesses (each 16-bit) by the VMEbus interface.

Example:

```plaintext
? MEM
Data bus width is set to 32 bits

? MEM 16

? MEM
Data bus width is set to 16 bits

? MEM 32

? MEM
Data bus width is set to 32 bits

? _
```
6.5.11  RUNINRAM – Run VMEPROM in RAM

Format

**RUNINRAM destination-address**

This command provides an easy way to copy the VMEPROM software from the system flash memory into the DRAM to run it there. First, the binary image of VMEPROM will be copied to the specified *destination-address*, then all absolute addresses of the image will be relocated. Finally, VMEPROM will completely be restarted at its new location.

Automatic copy

It is possible to let VMEPROM automatically copy its image into RAM after reset. After copying, the image is located at the end of memory and VMEPROM runs there.

To enable automatic copy use the FGA Boot **SETUP** command to set the Application Flags to 0001<sub>16</sub> (see section 8.2.12 “SETUP – Change Initialization Values” on page 193). Per default VMEPROM runs in the system PROM.

**IMPORTANT**

VMEPROM can not use memory beyond its own base address. If, for example, it is located at address 0020.0000<sub>16</sub>, VMEPROM can only use the memory range from 0000.0000<sub>16</sub> to 0020.0000<sub>16</sub>. Please see also the opposite command **RUNINROM**.

Example:

```
? LT
  task  pri  tm  ev1/ev2  size  pc  tcb  eom  ports
  *0/0  64  1  7868 FF027876 00007000 007B6000 1/1/0/0/0/0

? RUNINRAM 200000
  Copying program from $FF000000 to $00200000..<0024F18E
  Relocating program in new area
  Restarting VMEPROM ..........
  ************************************************************************
  * * V M E P R O M *
  * SYS68K/CPU-60  Version X.YZ  dd-mm-yy *
  * (c) FORCE Computers and Eyring Research *
  * *
  ************************************************************************

? LT
  task  pri  tm  ev1/ev2  size  pc  tcb  eom  ports
  *0/0  64  1  1726 00227876 00007000 001B6800 1/1/0/0/0/0

? _
```
6.5.12 RUNINROM – Run VMEPROM in ROM

Format

RUNINROM

This command restarts VMEPROM in the system flash memory.

Example:

```
? LT
task pri tm ev1/ev2 size pc tcb eom ports
*0/0 64 1 1726 00227876 00007000 001B6800 1/1/0/0/0/0

? RUNINROM
Restarting VMEPROM in ROM .........
******************************************************************
* *
* V M E P R O M *
* SYS68K/CPU-60 Version X.YZ dd-mmm-yy *
* (c) FORCE Computers and Eyring Research *
* *
******************************************************************

? LT
task pri tm ev1/ev2 size pc tcb eom ports
*0/0 64 1 7868 FF027876 00007000 007B6000 1/1/0/0/0/0

? 
```

6.5.13 SELFTEST – Perform On-Board Selftest

Format

SELFTEST

This command performs a test of the on-board functions of the CPU board. It can only run if no other tasks are created. If there are any other tasks, no self-test will be made and an error will be reported. SELFTEST tests the memory of the CPU board and all devices on the board.

The following tests are performed in this order:

1. I/O test

   This function tests the DMA controller, the SCRIPTS processor, and the interrupts of the SCSI controller. The floppy disk controller will be checked if it can be initialized. Then the access to the registers of the PCnet Ethernet controller will be tested. CIO1 and CIO2 will be tested if they are able to generate vector interrupts via a timer.

   If tests fail, error messages will be printed to state the type of fault.

2. Memory test on the memory of the current task
The following procedures are performed:

- Byte test
- Word test
- Longword test

All passes of the memory test perform pattern reading and writing as well as bit shift tests. If an error occurs while writing to or reading from the memory, it will be reported. Dependent on the size of the main memory, this test may last a different amount of time (count about one minute per Megabyte).

3. Clock test

If the CPU does not receive timer interrupts from the CIO1 Z8536, an error will be displayed. This ensures that VMEPROM can initialize the CIO1 Z8536 properly and the external interrupts from the CIO are working.

**IMPORTANT** During this process all contents of the memory are cleared.

**Example:**

```
? SELFTEST
VMEPROM Hardware Selftest
---------------------------
I/O test .......... passed
Memory test ...... passed
Clock test ...... passed

? _
```

### 6.6 Installing a New Hard Disk (Using `FRMT` and `INIT`)

This section provides an example how to use the `FRMT` and the `INIT` command to install a new hard disk.

The `FRMT` command of VMEPROM may be used

- to set all hard disk parameters,
- to format the Winchester,
- and to divide the disk into logical partitions.

Before starting the `FRMT` command the number of the last logical block of the Winchester must be known. The number of physical blocks per track must be 32, the number of bytes per sector must be 256.
The number of heads and the number of cylinders may be calculated by using the following equation:

\[(\text{# of heads}) \times (\text{# of cylinders}) \times (\text{blocks/track}) = \text{# of last logical block}\]

**IMPORTANT**

The SCSI ID must be 0, 1, or 2 and the maximum number of heads is 16. The number of large and floppy partitions can be defined by the user.

**Example:**

Formatting a CDC 94211-5 Winchester

```plaintext
? FRMT
68K PDOS Force Disk Format Utility
Possible Disk Controllers in this System are:
Controller #1 is not defined
Controller #2 is a FORCE WFC-1
Controller #3 is a FORCE ISCSI-1
Controller #4 is an onboard SCSI
Controller #5 is not defined
Controller #6 is a FORCE IBC
Drives that are currently defined in system are:
F0 is controller #4, drive select $82
F1 is controller #4, drive select $83
W0 is controller #4, drive select $00
All not named drives are undefined

Select Menu: W, W0-W15=Winch; F, F0-F8=Floppy; Q=Quit
Select Drive: W
W0 Main Menu: 1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ P)Togl Q)Quit
Command: 1

W0 Parameters Menu: A)lter, D)isplay, R)ead file, Q)uit
Command: A
  # of Heads = 10
  # of Cylinders = 1022
  Physical Blocks per Track = 32
  Physical Bytes per Block = 256
  Shipping Cylinder = 0
  Step rate = 0
  Reduced write current cyl = 0
  Write Precompensate cyl = 0

Current Winch Drive 0 Parameters:
  # of Heads = 10
  # of Cylinders = 1022
  Physical Blocks per Track = 32
  Physical Bytes per Block = 256
  Shipping Cylinder = 0
  Step rate = 0
  Reduced write current cyl = 0
  Write Precompensate cyl = 0`
```
Installing a New Hard Disk (Using FRMT and INIT)

WO Parameters Menu: A)lter, D)isplay, R)ead file, Q)uit
Command: Q

WO Main Menu: 1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ
Command: 3
Sector Interleave = 0
Physical Tracks to FORMAT = 0,10219
Ready to FORMAT Winchester Drive 0 ? Y

Sector Interleave Table: 0,1,2,3,4,5,6,7,8,9,10,11,12,
13,14,15,16,17,18,19,20,21,22,
23,24,25,26,27,28,29,30,31

Issuing Format Drive Command.
FORMAT SUCCESSFUL !

WO Main Menu: 1)Parm 2)BadT 3)Form 4)Veri 5)Part 6)Writ
Command: 5

WO Partitions Menu: A)lter, D)isplay, R)ecalc, Q)uit
Command: A

# of Large partitions = 6
# of Floppy Partitions = 15
First track for PDOS Parts = 0
Last track for PDOS Parts = 10219
First PDOS disk # = 2

Current Winch Drive 0 Partitions:
# of Large partitions = 6
# of Floppy Partitions = 15
First track for PDOS Parts = 0
Last track for PDOS Parts = 10219
First PDOS disk # = 2
Total # of Logical Tracks = 10220

<table>
<thead>
<tr>
<th>Disk #</th>
<th>Logical Trks</th>
<th>Physical Trks</th>
<th>PDOS sectors</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Base,Top</td>
<td>Base,Top</td>
<td>Total/(boot)</td>
</tr>
<tr>
<td>2</td>
<td>0,1502</td>
<td>0,1502</td>
<td>48064/47872</td>
</tr>
<tr>
<td>3</td>
<td>1503,3005</td>
<td>1503,3005</td>
<td>48064/47872</td>
</tr>
<tr>
<td>4</td>
<td>3006,4508</td>
<td>3006,4508</td>
<td>48064/47872</td>
</tr>
<tr>
<td>5</td>
<td>4509,6011</td>
<td>4509,6011</td>
<td>48064/47872</td>
</tr>
<tr>
<td>6</td>
<td>6012,7514</td>
<td>6012,7514</td>
<td>48064/47872</td>
</tr>
<tr>
<td>7</td>
<td>7515,9017</td>
<td>7515,9017</td>
<td>48064/47872</td>
</tr>
<tr>
<td>9</td>
<td>9018,9097</td>
<td>9018,9097</td>
<td>2528/2336</td>
</tr>
<tr>
<td>10</td>
<td>9098,9177</td>
<td>9098,9177</td>
<td>2528/2336</td>
</tr>
<tr>
<td>11</td>
<td>9178,9257</td>
<td>9178,9257</td>
<td>2528/2336</td>
</tr>
<tr>
<td>12</td>
<td>9258,9337</td>
<td>9258,9337</td>
<td>2528/2336</td>
</tr>
<tr>
<td>13</td>
<td>9338,9417</td>
<td>9338,9417</td>
<td>2528/2336</td>
</tr>
<tr>
<td>14</td>
<td>9418,9497</td>
<td>9418,9497</td>
<td>2528/2336</td>
</tr>
<tr>
<td>15</td>
<td>9498,9577</td>
<td>9498,9577</td>
<td>2528/2336</td>
</tr>
<tr>
<td>16</td>
<td>9578,9657</td>
<td>9578,9657</td>
<td>2528/2336</td>
</tr>
<tr>
<td>17</td>
<td>9658,9737</td>
<td>9658,9737</td>
<td>2528/2336</td>
</tr>
<tr>
<td>18</td>
<td>9738,9817</td>
<td>9738,9817</td>
<td>2528/2336</td>
</tr>
<tr>
<td>19</td>
<td>9818,9897</td>
<td>9818,9897</td>
<td>2528/2336</td>
</tr>
<tr>
<td>20</td>
<td>9898,9977</td>
<td>9898,9977</td>
<td>2528/2336</td>
</tr>
<tr>
<td>21</td>
<td>9978,10057</td>
<td>9978,10057</td>
<td>2528/2336</td>
</tr>
<tr>
<td>22</td>
<td>10058,10137</td>
<td>10058,10137</td>
<td>2528/2336</td>
</tr>
<tr>
<td>23</td>
<td>10138,10217</td>
<td>10138,10217</td>
<td>2528/2336</td>
</tr>
</tbody>
</table>
After formatting the disk all logical partitions must be initialized using the \texttt{INIT} command.

\textbf{Example:}

Initialize the large logical partition (number 2)

\begin{verbatim}
? INIT
Enter Disk # : 2
Directory Entries : 1024
Number of sectors : 47776
Disk Name : SYSTEM
Init: Disk # 2
  Directory entries: 1024
  Number of sectors: 47776
  Disk name: SYSTEM
Initialize disk ? Y
?
\end{verbatim}
7 Appendix to VMEMPRO

7.1 Driver Installation

This appendix summarizes the changes to be made to the default setup of additional VMEbus boards so that they are VMEMPRO-compatible. Drivers described in the following sub-sections are available in ROM, but not all are installed. However, drivers for all on-board devices are automatically installed.

**INSTALL** command

To install a driver use the **INSTALL** command.

**IMPORTANT**

Software version dependent addresses

The addresses given in the examples of this section are only example UART and Disk Driver addresses. They may vary across software versions.

**Current configuration**

To view the current configuration issue the **install** command as shown in the following example:

```
? INSTALL ?
THE FOLLOWING UARTS AND DISK DRIVER ARE ALREADY IN EPROM:

<table>
<thead>
<tr>
<th>Device Type</th>
<th>Board Type</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>DISK DRIVER</td>
<td>FORCE IBC/ME</td>
<td>$FF029500</td>
</tr>
<tr>
<td>DISK DRIVER</td>
<td>FORCE ISCSI-1</td>
<td>$FF029700</td>
</tr>
<tr>
<td>DISK DRIVER</td>
<td>FORCE SCSI CPU-60</td>
<td>$FF02A300</td>
</tr>
<tr>
<td>DISK DRIVER</td>
<td>FORCE WFC-1</td>
<td>$FF02BA00</td>
</tr>
<tr>
<td>UART DRIVER</td>
<td>FORCE IBC/ME</td>
<td>$FF02E200</td>
</tr>
<tr>
<td>UART DRIVER</td>
<td>FORCE ISIO-1/2</td>
<td>$FF02E400</td>
</tr>
<tr>
<td>UART DRIVER</td>
<td>FORCE CPU-60/28530</td>
<td>$FF02E900</td>
</tr>
<tr>
<td>UART DRIVER</td>
<td>FORCE SIO-1/2</td>
<td>$FF02EE00</td>
</tr>
</tbody>
</table>
```

7.1.1 VMEbus Memory

In general every FORCE memory board can be used together with VMEMPRO.

In order to use a memory board within the tasking memory of VMEMPRO the base address must be set correctly. That means that the board base addresses of any additional memory boards must be set to be contiguous to the on-board memory.
7.1.2 SYS68K/SIO-1/2

By default, the two serial I/O boards SYS68K/SIO-1/2 are set to the VME base address \( B0.0000_{16} \). VMEPROM expects the first SIO-1/2 boards at \( FCB0.0000_{16} \). This is in the standard VME address range (A24, D16, D8) with the address \( B0.0000_{16} \).

The address modifier decoder (AM-Decoder) of the SIO-1/2 boards must be set to:

- Standard Privileged Data Access
- Standard Non-Privileged Data Access

Please refer to the SIO User's Manual for the setup. If a second SIO-1/2 board will be used, the base address must be set to \( FCB0.0200_{16} \). The AM-decoder setup described above has to be used again. Please refer to the SIO User's Manual for the address setup of the second SIO board.

To install driver

To install the SIO-1/2 board driver use the `install` command with the appropriate address (see “Software version dependent addresses” on page 151):

```
? INSTALL U2,$FF02EE00
```

To install a port

To install one of the ports of the SIO boards in VMEPROM use the `BP` command. The SIO-1/2 boards use the driver type 2.

**IMPORTANT**

The hardware configuration must be detected before a port can be installed. This can be done by using the `CONFIG` command or by setting a front panel switch on the CPU board and pressing reset.

**Example:**

To install the first port of a SIO board with a 9600 baud rate as port number 3 enter the following:

```
? BP 3,9600,2,$FCB00000
```
Simultaneous use of SIO-1/2 and ISIO-1/2

VMEPROM supports up to two serial I/O boards. These can be either the SIO-1/2 board, the ISIO-1/2 board, or a mixture of both.

**IMPORTANT**

The first board of every type must be set to the first base address. If one SIO-1 board and one ISIO-1 board are used, the base address of the boards must be set to:

- $\text{FCB0.}0000_{16}$ for SIO-1
- $\text{FC96.}0000_{16}$ for ISIO-1

### 7.1.3 SYS68K/ISIO-1/2

By default, the serial I/O boards SYS68K/ISIO-1/2 are set to the address $96.0000_{16}$ in the standard VME address range. VMEPROM awaits this board at this address ($\text{FC96.}0000_{16}$ for the CPU-60); changes to the default setup are not necessary. An optional second board may be used. In this case, the address must be set to $98.0000_{16}$. For a description of the base address setup, read the *SYS68K/ISIO-1/2 User's Manual*. 

---

**Table 67** Base addresses of SIO-1/2 ports

<table>
<thead>
<tr>
<th>SIO board</th>
<th>Port #</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>first</td>
<td>1</td>
<td>FCB0.0000_{16}</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>FCB0.0040_{16}</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>FCB0.0080_{16}</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>FCB0.00C0_{16}</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>FCB0.0100_{16}</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>FCB0.0140_{16}</td>
</tr>
<tr>
<td>second</td>
<td>1</td>
<td>FCB0.0200_{16}</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>FCB0.0240_{16}</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>FCB0.0280_{16}</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>FCB0.02C0_{16}</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>FCB0.0300_{16}</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>FCB0.0340_{16}</td>
</tr>
</tbody>
</table>
To install driver

To install the ISIO-1/2 board driver use the install command with the appropriate address (see “Software version dependent addresses” on page 151):

```plaintext
? INSTALL U3,$FF02E400
```

To install a port

To install one of the ports of an ISIO board in VMEPROM use the BP command. The ISIO-1/2 boards are driver type 3.

**IMPORTANT**

The hardware configuration must be detected before a port can be installed. This can be done by using the CONFIG command or by setting a front panel switch on the CPU board and pressing reset.

**Example:**

To install the first port of an ISIO board with a 9600 baud rate as port number 3 enter the following:

```plaintext
? BP 3,9600,3,$FC968000
```

### Table 68 Base addresses of ISIO-1/2 ports

<table>
<thead>
<tr>
<th>ISIO board</th>
<th>Port #</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>first</td>
<td>1</td>
<td>FC96.8000_{16}</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>FC96.8020_{16}</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>FC96.8040_{16}</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>FC96.8060_{16}</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>FC96.8080_{16}</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>FC96.80A0_{16}</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>FC96.80C0_{16}</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>FC96.80E0_{16}</td>
</tr>
</tbody>
</table>
Simultaneous use of SIO-1/2 and ISIO-1/2

VMEPROM supports up to two serial I/O boards. These can be either the SIO-1/2 board, the ISIO-1/2 board, or a mixture of both.

**IMPORTANT**

The first board of every type must be set to the first base address. If one SIO-1 board and one ISIO-1 board are used, the base address of the boards must be set to:

- **FCB0.0000** for SIO-1
- **FC96.0000** for ISIO-1

### 7.1.4 SYS68K/WFC-1 Disk Controller

VMEPROM supports up to two floppy disk drives and three Winchester disk drives together with the WFC-1 disk controller. The floppy drives must be jumpered to drive select 3 and 4. VMEPROM accesses them as disk number 0 and 1. The floppy drives are automatically installed when a WFC-1 controller is detected by the `CONFIG` command or after reset when the front-panel switch of the CPU board is set to detect the hardware configuration. Only double-sided and double-density floppy drives which support 80 tracks/side can be used. The step rate is 3 ms.

The Winchester drives are not installed automatically. The `FRMT` command must be used for defining the following factors:

- The physical drive structure (i.e. number of heads, number of cylinders, drive select number, etc.)
- The bad block of the Winchester drive
- The partitions to be used

<table>
<thead>
<tr>
<th>ISIO board</th>
<th>Port #</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>second</td>
<td>1</td>
<td>FCB0.0000&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>FCB0.0200&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>FCB0.0400&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>FCB0.0600&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>FCB0.0800&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>FCB0.0A00&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>FCB0.0C00&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>FCB0.0E00&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
</tbody>
</table>
If this setup is done once for a particular drive, the data is stored in the first sector of the Winchester and automatically loaded when the disk controller is installed in VMEPROM.

To install driver

To install the driver for the WFC-1 use the **install** command with the appropriate address (see “Software version dependent addresses” on page 151):

```
? INSTALL W,$FF02BA00
```

The default base address of the WFC-1 controller must be set to FCB0.1000. That means the address comparison for 32-bit address has to be enabled and the setup of the most significant 8 addresses must be jumpered.

VMEPROM supports termination interrupt of the WFC-1 controller. If you want to use the WFC-1 in combination with interrupts, the corresponding jumper must be set to enable the interrupt.

For a detailed description of the address setup and termination interrupt, refer to the data sheet of the WFC-1 controller.

### 7.1.5 SYS68K/ISCSI-1 Disk Controller

VMEPROM supports up to two floppy disk drives and three Winchester disk drives together with the ISCSI-1 disk controller. The floppy drives must be jumpered to drive select 3 and 4. VMEPROM accesses them as disk number 0 and 1. The floppy drives are installed automatically when an ISCSI-1 controller is detected by the `CONFIG` command or after pressing reset when the front panel switch of the CPU board is set to detect the hardware configuration. Only double-sided and double-density floppy drives which support 80 tracks/side can be used. The used step rate is 3 ms. The Winchester drives are not installed automatically. The VMEPROM `FRMT` command must be used for defining the following factors:

- The physical structure of the drive (i.e., number of heads, number of cylinders, drive select number, etc.)
- The bad block of the Winchester drive
- The partitions to be used

If this setup is done once for a particular drive, the data is stored in the first sector of the Winchester and automatically loaded when the disk controller is installed in VMEPROM.
To install driver

To install the ISCSI-1 driver use the **install** command with the appropriate address (see “Software version dependent addresses” on page 151):

```
? INSTALL W, $FF029700
```

The default base address of the ISCSI-1 controller is \( A0.0000_{16} \) in the standard VME address range. This is the address \( FCA0.0000_{16} \) for the CPU board. To this setup no changes have to be made. The ISCSI-1 driver uses interrupts by default. This cannot be disabled. Please make sure that the interrupt daisy chain is closed so that the controller can work properly.

### 7.1.6 Local SCSI Controller

VMEPROM supports up to three Winchester disk drives together with the local SCSI Controller. The Winchester drives are not installed automatically.

The VMEPROM **FRMT** command must be used for defining the following factors:

- The physical structure of the drive (i.e., number of heads, number of cylinders, drive select number, etc.)
- The bad blocks of the Winchester drive
- The partitions to be used

If this setup is done once for a particular drive, the data is stored in the first sector of the Winchester and automatically loaded when the disk controller is installed in VMEPROM. When viewing the VMEPROM banner, the driver for the local SCSI controller is already installed. This driver needs memory for hashing. The storage for the hashing buffers is allocated at the end of memory.
7.2 S-Record Formats

S-record types 8 types of S-records have been defined to accommodate the needs of encoding, transportation and decoding functions:

Table 69 Types of S-record format modules and VMEPROM support

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>S0</td>
<td>The header record for each block of S-records</td>
</tr>
<tr>
<td>S1</td>
<td>A record containing code/data and the 2-byte address at which the code/data is to reside</td>
</tr>
<tr>
<td>S2</td>
<td>A record containing code/data and the 3-byte address at which the code/data is to reside</td>
</tr>
<tr>
<td>S3</td>
<td>A record containing code/data and the 4-byte address at which the code/data is to reside</td>
</tr>
<tr>
<td>S5</td>
<td>A record containing the number of S1, S2 and S3 records transmitted in a particular block. The count appears in the address field. There is no code/data field.</td>
</tr>
<tr>
<td>S7</td>
<td>A termination record for a block of S3 records. The address field may optionally contain the 4-byte address of the instruction to which control is to be passed. There is no code or data field.</td>
</tr>
<tr>
<td>S8</td>
<td>A termination record for a block of S2 records. The address field may optionally contain the 3-byte address of the instruction to which control is to be passed. There is no code or data field.</td>
</tr>
<tr>
<td>S9</td>
<td>A termination record for a block of S1 records. The address field may optionally contain the 2-byte address of the instruction to which control is to be passed.</td>
</tr>
</tbody>
</table>

General S-record type use Only one termination record is used for each block of S-records. In general, S7 and S8 records are only used when control is to be passed to a 3- or 4-byte address. Normally, only one header record is used, but it is also possible that multiple header records occur.
Example:

```
S21402000000004440002014660000CB241F8044CB1
S214020010203C00000020E428110C1538066FA487AE4
S21402002001021DF0008487A001221DF000C4E750E
S21402003021FC425553200030600821FC4144452C2
XX------------------------ Check-sum
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX--- Data
0200XX----------------------------------- 24-bit Address
14----------------------------------------- Byte Count
S2------------------------------------------- Record Type

S9030000FC

FC---------------------------------------- Check-sum
0000-------------------------------------- Data
03----------------------------------------- Byte Count
S9------------------------------------------- Record Type
```

### 7.3 System RAM Definitions

```c
/* SYRAM.H -- DEFINITION OF SYRAM BLOCK OF MEMORY  
05-Jan-88 Revised to correspond to PDOS 3.3  
BRIAN C. COOPER, EYRING RESEARCH INSTITUTE, INC.  
Copyright 1985-1988 */
#define NT 64 /* number of tasks */
#define NM ((NT+3)&0xFC) /* number of task messages */
#define NP 16 /* number of task message pointers */
#define ND ((NT+3)&0xFC) /* number of delay events */
#define NC 8 /* number of active channel buffers */
#define NF 64 /* number of file slots */
#define NU 15 /* number of I/O UART ports */
#define IZ 6 /* input buffer size (2^p2p. */
#define MZ 0x4000000 /* maximum memory size */
#define TZ 64 /* task message size */
#define NTB NT
#define NTM NM
#define NTP NP
#define NCB NC
#define NFS NF
#define NEV ND
#define NIE (ND/2)
#define NPS (NU+1)
#define P2P IZ
#define MMZ MZ
#define TMZ TZ
```
```c
#define IMK (0xFF>>(8-P2P))  /* input buffer wrap around mask */
#define NCP ((1<<P2P)+2)  /* (# characters/port) + 2 */
#define MPZ 2048  /* memory page size */
#define MBZ (MMZ/MPZ)  /* memory bitmap size */
#define NMB (MBZ/8)  /* number of map bytes */
#define FSS 38  /* file slot size */
#define TQB 2  /* TCB index */
#define TQM (TQB+4)  /* map index */
#define TQE (TQM+2)  /* event #1 / event #2 */
#define TQS (TQB+2)  /* scheduled event */
#define TBZ (TQS+2)  /* TASK entry size */
#define BPS 256  /* bytes per sector */
#define RRD 4  /* number of RAM disks */

struct SYRAM{  /* address of bios rom */
    char _bios;    /* mail array address */
    char _mail;    /* ram disk # */
    unsigned int _rdkn; /* ram disk size */
    unsigned int _rdks; /* ram disk address */
    char _rdka;    /* basic present flag */
    char _dflg;    /* directory flag */
    int _f681;    /* 68000/68010 flag */
    char _sram;    /* run module B$SRAM */
    int spare1;    /* reserved for expansion */
    int _fcnt;    /* fine counter */
    char _smon;    /* 32 bit counter */
    unsigned char _sday; /* month */
    unsigned char _smon; /* day */
    unsigned char _syrs[2]; /* year */
    unsigned char _shrs; /* hours */
    unsigned char _smin; /* minutes */
    unsigned char _sssec[2]; /* seconds */
    unsigned char _patb[16]; /* input port allocation table */
    char _brkf[16]; /* input break flags */
    char _f8bt[16]; /* port flag bits */
    char _utyp[16]; /* port uart type */
    char _urat[16]; /* port rate table */
    char _evtb[10]; /* 0-79 event table */
    char _evto[2]; /* 80-95 output events */
    char _evti[2]; /* 96-111 input events */
    char _evts[2]; /* 112-127 system events */
    char _evt[2][16]; /* task 128 events */
    long _evtm[4]; /* events 112-115 timers */
    long _bclk; /* clock adjust constant */
    char _tltp; /* task list pointer */
    char _utcb; /* user tcb ptr */
    int _suim; /* supervisor interrupt mask */
    int _usim; /* user interrupt mask */
    char _sptn; /* spawn task no. (** must be even **)*/
    char _utim; /* user task time */
    char _tpry; /* task priority (** must be even **)*/
    char _tskn; /* current task number */
    char spare2; /* reserved */
    char _tqux; /* task queue offset flag/no */
    char _tlck[2]; /* task lock/reschedule flags */
}```
/*0C0*/ char _e122; /* batch task # */
/*0C1*/ char _e123; /* spooler task # */
/*0C2*/ char _e124;
/*0C3*/ char _e125;
/*0C4*/ long _cksm; /* system checksum */
/*0C8*/ int _pnod; /* pnet node # */
/*0CA*/ char bser[6]; /* bus error vector */
/*0DD*/ char iler[6]; /* illegal vector */
/*0DE*/ char ccnt[16]; /* control C count */
/*0E6*/ char *wind; /* window id's */
/*0EA*/ char *wadr; /* window addresses */
/*0EE*/ char *chot; /* input stream */
/*0F2*/ char *cht; /* output stream */
/*0F6*/ char *iord; /* i/o redirect */
/*0FA*/ char _fect; /* file expand count */
/*0FB*/ char *pidn; /* processor ident byte */
/*0FC*/ long *begn; /* abs addr of K1$BEGIN table */
/*100*/ int rwcl[14]; /* port row/col 1..15 */
/*11C*/ char *opip[15]; /* output port pointers 1..15 */
/*158*/ char *uart[16]; /* uart base addresses 1..15 */
/*198*/ long _mapb; /* memory map bias */
/*
* the following change with different configurations:
*/
/* configuration for VMEPROM is defined to: */
/*
* NT = 64, NF = 64, MZ = $400000 */
/*
*/
/* NOTE: the offset on top of each line is calculated only for this */
/* configuration */
/*
*019C*/ char _maps[NMB]; /* system memory bitmap */
*/
/*119C*/ char _port[(NPS-1)*NCP]; /* character input buffers */
/*157A*/ char _iout[(NPS-1)*NCP]; /* character output buffers */
/*1958*/ char rdtb[16]; /* redirect table */
/*1968*/ int _tque[NTB+1]; /* task queue */
/*19EA*/ char _lst[NTB+TBZ]; /* task list */
/*1DEA*/ char _stev[NTB+32]; /* task schedule event table */
/*25EA*/ long _tmf[NTM]; /* to/from/INDEX.W */
/*
*26EA*/ char _mbf[TMZ+NTM]; /* task message buffers */
/*36EA*/ char _msp[NTF*6]; /* task message pointers */
/*374A*/ char _dsq[2+8+NIE*10]; /* delay event insert queue */
/*3894*/ char _devt[2+NEV*10]; /* delay events */
/*3B16*/ int _bsct[32]; /* basic screen command table */
/*3B56*/ int _xchi[NCB]; /* channel buffer queue */
/*3B66*/ char _xchb[NCB*BPS]; /* channel buffers */
/*4366*/ char _xfsl[NFS*FSS]; /* file slots */
/*4CE6*/ char _l2lk; /* level 2 lock {file prims, evnt 120} */
/*4CE7*/ char _l3lk; /* level 3 lock {disk prims, evnt 121} */
/*4CE8*/ long _drv1; /* driver link list entry point */
/*4CEC*/ long _ultl; /* utility link list entry point */
/*4CF0*/ int _rdkl[NRD*4 + 1]; /* RAM disk list */
);
Task Control Block Definitions

# define MAXARG 10     /* max argument count of the cmd line */
# define MAXBP 10      /* max 10 breakpoints */
# define MAXNAME 5     /* max 5 names in name buffer */
# define TMAX 64       /* Max number of tasks */
# define ARGLEN 20     /* maximum argument length */

/* special system flags for VMEPROM */

# define SOMEREG 0x0001 /* display only PC,A7,A6,A5 */
# define T_DISP 0x0002  /* no register display during trace(TC>1) */
# define T_SUB 0x0004   /* trace over subroutine set */
# define T_ASUB 0x0008  /* trace over subroutine active */
# define T_RANG 0x0010  /* trace over range set */
# define REG_INI 0x0020 /* no register initialization if set */
# define RE_DIR 0x0040  /* output redirection into file and */
                        /* console at the same time */

/* the registers are stored in the following order: */

# define VBR 0
# define SFC 1
# define DFC 2
# define CACR 4
# define PC 5
# define SR 6
# define USTACK 7
# define SSTACK 8
# define MSTACK 9
# define D0 10 /* 10-17 = D0-D7 */
# define A0 18 /* 18-24 = A0-A6 */

# define N_REGS 25

# define BYTE unsigned char
# define WORD unsigned int
# define LWORD unsigned long

struct TCB {

    /*000*/ char _ubuf[256]; /* 256 byte user buffer */
    /*100*/ char _clb[80];   /* 80 byte monitor command line buffer */
    /*150*/ char _mwb[32];   /* 32 byte monitor parameter buffer */
    /*170*/ char _mpb[60];   /* monitor parameter buffer */
    /*1AC*/ char _cob[8];    /* character out buffer */
    /*1B4*/ char _swb[508];  /* system work buffer/task pdos stack */
    /*3B0*/ char *_tsp;      /* task stack pointer */
    /*3B4*/ char *_kil;      /* kill self pointer */
    /*3B8*/ long _sfp;       /* RESERVED FOR INTERNAL PDOS USE */
    /*3BC*/ char _svf;       /* save flag -- 68881 support (x881) */
    /*3BD*/ char _iff;       /* RESERVED FOR INTERNAL PDOS USE */
    /*3BE*/ long _trp[16];   /* user TRAP vectors */
    /*3FE*/ long _zdv;       /* zero divide trap */
    /*402*/ long _chk;       /* CHK instruction trap */
    /*406*/ long _trv;       /* TRAPV Instruction trap */
/*40A*/ long _trc; /* trace vector */
/*40E*/ long _fpa[2]; /* floating point accumulator */
/*416*/ long *_fpe; /* fp error processor address */
/*41A*/ char *clp; /* command line pointer */
/*41E*/ char *_bum; /* beginning of user memory */
/*422*/ char *eum; /* end user memory */
/*426*/ char *ead; /* entry address */
/*42A*/ char *imp; /* internal memory pointer */
/*42E*/ int _acl; /* assigned input file ID */
/*430*/ int _acl2; /* assigned input file ID's */
/*434*/ int _len; /* last error number */
/*438*/ byte _flg; /* task flags (bit 8=command line echo) */
/*43C*/ BYTE slv; /* directory level */
/*43E*/ char *_fpe; /* fp error processor address */
/*442*/ char *ext; /* XEXT address */
/*446*/ char *err; /* XERR address */
/*44A*/ char _cmd; /* command line delimiter */
/*44B*/ BYTE _tid; /* task id */
/*44C*/ char _ecf; /* echo flag */
/*44E*/ char _cnt; /* output column counter */
/*450*/ char _spu; /* spooling unit mask */
/*451*/ BYTE _unt; /* output unit mask */
/*452*/ char _ulp; /* unit 1 port # */
/*453*/ char _u2p; /* unit 2 port # */
/*454*/ char _u4p; /* unit 4 port # */
/*455*/ char _u8p; /* unit 8 port # */
/*456*/ char _spare2[26]; /* reserved for system use */

/***********************************************************************
/* VMEPROM variable area area */
/***********************************************************************
/*470*/ char linebuf[82]; /* command line buffer */
/*4C2*/ char alinebuf[82]; /* alternate line buffer */
/*514*/ char cmdline[82]; /* alternate cmdline for XGNP */
/*566*/ int allargs, gotargs; /* argc save and count for XGNP */
/*56A*/ int argc; /* argument counter */
/*56C*/ char *argv[MAXARG]; /* pointer to arguments of the cmd line */
/*594*/ char *odir, *idir; /* I/O redirection args from cmd line */
/*59C*/ int iport, oport; /* I/O port assignments */
/*5A0*/ char *ladr; /* holds pointer to line in_mwb */
/*5A4*/ LWORD offset; /* base memory pointer */
/*5A8*/ int bpcnt; /* num of defined breakpoints */
/*5AA*/ LWORD bpadr[MAXBP]; /* breakpoint address */
/*5D2*/ WORD bpinst[MAXBP]; /* breakpoint instruction */
/*5E6*/ char bpcmd[MAXBP][1]; /* breakpoint command */
Task Control Block Definitions

/*654*/ WORD bpoct[MAXBP]; /* # of times the breakpoint should be */ /* skipped */
/*668*/ WORD bpcocc[MAXBP]; /* # of times the breakpoint is already */ /* skipped */
/*67C*/ LWORD bptadr; /* temp. breakpoint address */
/*680*/ WORD bptinst; /* temp. breakpoint instruction */
/*682*/ WORD bptocc; /* # of times the temp. breakpoint should */ /* be skipped */
/*684*/ WORD bptcocc; /* # of times the temp. breakpoint is */ /* already skipped */
/*686*/ char bptcmd[11]; /* temp. breakpoint command */
/*691*/ char bptcmd[11]; /* temp. breakpoint command */
/*692*/ char outflag; /* output messages (yes=1,no=0) */
/*6A9*/ char namebn[MAXNAME][8]; /* Name buffer, name */
/*6B0*/ char namebd[MAXNAME][40]; /* Name buffer, data */
/*721*/ WORD errcnt; /* error counter for test .. */
/*734*/ LWORD times,timee; /* start/end time */
/*7C0*/ WORD preg[N_REGS]; /* storage area of processor regs */
/*7F0*/ WORD tflag; /* trace active flag */
/*7F2*/ WORD tcount; /* trace count */
/*7F4*/ WORD tacount; /* active trace count */
/*7F6*/ WORD bpact; /* break point active flag */
/*7F8*/ LWORD savesp; /* save VMEprom stack during GO/T etc */
/*7FC*/ char VMEMSP[202]; /* Master stack, handle w/ care */
/*8C0*/ char VMESSP[802]; /* supervisor stack, handle w/ care */
/*8E8*/ char VMESPUP[802]; /* vmem internal user stack */
/*F0A*/ LWORD _f_fpreg[3*8]; /* floating point data regs */
/*F6A*/ LWORD _f_fpcr; /* FPCR reg */
/*F6C*/ LWORD _f_fpsr; /* FPSR reg */
/*F72*/ LWORD _f_fpiar; /* FPIAR reg */
/*F76*/ BYTE _f_save[0x3c]; /* FPSAVE for null and idle */
/*F84*/ BYTE _cleon[2]; /* clear to end of screen parameter */
/*F86*/ BYTE _cleol[2]; /* clear to end of line parameters */
/*F8C*/ char _u_prompt[10]; /* user defined prompt sign */
/*FC0*/ long _c_save; /* save Cache control register */
/*FC4*/ long _exe_cnt; /* execution count */
/*FC8*/ BYTE _nokill; /* kill task with no input port */
/*FC9*/ BYTE _u_mask; /* unit mask for echo */
/*FCA*/ WORD sysflg; /* system flags used by VMEpROM */
/* FCC*/ LWORD _t_range[2]; /* start/stop PC for trace over range */
/*FD4*/ LWORD _ex_regs; /* pointer to area for saved regs */
/*FD8*/ BYTE _sparend[0x1000-0xFD8]; /* make tcb size $1000 bytes */
char _tbe[0]; /* task beginning */
}
### 7.5 Interrupt Vector Table of VMEPROM

<table>
<thead>
<tr>
<th>Vector number</th>
<th>Vector address</th>
<th>Assignment</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>00&lt;sub&gt;16&lt;/sub&gt;</td>
<td>000&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>1</td>
<td>01&lt;sub&gt;16&lt;/sub&gt;</td>
<td>004&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>2</td>
<td>02&lt;sub&gt;16&lt;/sub&gt;</td>
<td>008&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>3</td>
<td>03&lt;sub&gt;16&lt;/sub&gt;</td>
<td>00C&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>4</td>
<td>04&lt;sub&gt;16&lt;/sub&gt;</td>
<td>010&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>5</td>
<td>05&lt;sub&gt;16&lt;/sub&gt;</td>
<td>014&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>6</td>
<td>06&lt;sub&gt;16&lt;/sub&gt;</td>
<td>018&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>7</td>
<td>07&lt;sub&gt;16&lt;/sub&gt;</td>
<td>01C&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>8</td>
<td>08&lt;sub&gt;16&lt;/sub&gt;</td>
<td>020&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>9</td>
<td>09&lt;sub&gt;16&lt;/sub&gt;</td>
<td>024&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>10</td>
<td>0A&lt;sub&gt;16&lt;/sub&gt;</td>
<td>028&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>11</td>
<td>0B&lt;sub&gt;16&lt;/sub&gt;</td>
<td>02C&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>12</td>
<td>0C&lt;sub&gt;16&lt;/sub&gt;</td>
<td>030&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>13</td>
<td>0D&lt;sub&gt;16&lt;/sub&gt;</td>
<td>034&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>14</td>
<td>0E&lt;sub&gt;16&lt;/sub&gt;</td>
<td>038&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>15</td>
<td>0F&lt;sub&gt;16&lt;/sub&gt;</td>
<td>03C&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>16-23</td>
<td>10&lt;sub&gt;16&lt;/sub&gt;-17&lt;sub&gt;16&lt;/sub&gt;</td>
<td>040&lt;sub&gt;16&lt;/sub&gt;-05C&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>24</td>
<td>18&lt;sub&gt;16&lt;/sub&gt;</td>
<td>060&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>25</td>
<td>19&lt;sub&gt;16&lt;/sub&gt;</td>
<td>064&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>26</td>
<td>1A&lt;sub&gt;16&lt;/sub&gt;</td>
<td>068&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>27</td>
<td>1B&lt;sub&gt;16&lt;/sub&gt;</td>
<td>06C&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>28</td>
<td>1C&lt;sub&gt;16&lt;/sub&gt;</td>
<td>070&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>29</td>
<td>1D&lt;sub&gt;16&lt;/sub&gt;</td>
<td>074&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>30</td>
<td>1E&lt;sub&gt;16&lt;/sub&gt;</td>
<td>078&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>31</td>
<td>1F&lt;sub&gt;16&lt;/sub&gt;</td>
<td>07C&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>32-47</td>
<td>20&lt;sub&gt;16&lt;/sub&gt;-2F&lt;sub&gt;16&lt;/sub&gt;</td>
<td>080&lt;sub&gt;16&lt;/sub&gt;-0B&lt;sub&gt;C16&lt;/sub&gt;</td>
</tr>
<tr>
<td>48</td>
<td>30&lt;sub&gt;16&lt;/sub&gt;</td>
<td>0C0&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
<tr>
<td>Vector number</td>
<td>Vector address</td>
<td>Assignment</td>
</tr>
<tr>
<td>---------------</td>
<td>---------------</td>
<td>-------------------------------------------------</td>
</tr>
<tr>
<td>49</td>
<td>31_{16}</td>
<td>0C4_{16} FP inexact result</td>
</tr>
<tr>
<td>50</td>
<td>32_{16}</td>
<td>0C8_{16} FP divide by zero</td>
</tr>
<tr>
<td>51</td>
<td>33_{16}</td>
<td>0CC_{16} FP underflow</td>
</tr>
<tr>
<td>52</td>
<td>34_{16}</td>
<td>0D0_{16} FP operand error</td>
</tr>
<tr>
<td>53</td>
<td>35_{16}</td>
<td>0D4_{16} FP overflow</td>
</tr>
<tr>
<td>54</td>
<td>36_{16}</td>
<td>0D8_{16} FP signaling NAN</td>
</tr>
<tr>
<td>55</td>
<td>37_{16}</td>
<td>0DC_{16} FP unimplemented data type</td>
</tr>
<tr>
<td>56</td>
<td>38_{16}</td>
<td>0E0_{16} PMMU configuration</td>
</tr>
<tr>
<td>57</td>
<td>39_{16}</td>
<td>0E4_{16} PMMU illegal operation</td>
</tr>
<tr>
<td>58</td>
<td>3A_{16}</td>
<td>0E8_{16} PMMU access level violation</td>
</tr>
<tr>
<td>59</td>
<td>3B_{16}</td>
<td>0EC_{16} reserved (Unassigned)</td>
</tr>
<tr>
<td>60</td>
<td>3C_{16}</td>
<td>0F0_{16} Unimplemented effective address</td>
</tr>
<tr>
<td>61</td>
<td>3D_{16}</td>
<td>0F4_{16} Unimplemented integer instruction</td>
</tr>
<tr>
<td>62-63</td>
<td>3E_{16} - 3F_{16}</td>
<td>0F8_{16} - 0FC_{16} reserved (Unassigned)</td>
</tr>
<tr>
<td>64-75</td>
<td>40_{16} - 4B_{16}</td>
<td>100_{16} - 12C_{16} SIO-1/2 interrupt vectors, port #1 - 6</td>
</tr>
<tr>
<td>76-83</td>
<td>4C_{16} - 53_{16}</td>
<td>130_{16} - 14C_{16} ISIO-1/2 interrupt vectors, Port #1,2 - 15,16</td>
</tr>
<tr>
<td>84-118</td>
<td>54_{16} - 76_{16}</td>
<td>150_{16} - 1D8_{16} User defined vectors</td>
</tr>
<tr>
<td>119</td>
<td>77_{16}</td>
<td>1DC_{16} Disk interrupt vector (ISCSI-1)</td>
</tr>
<tr>
<td>120-191</td>
<td>78_{16} - BF_{16}</td>
<td>1EO_{16} - 2FC_{16} User defined vectors</td>
</tr>
<tr>
<td>192</td>
<td>C0_{16}</td>
<td>300_{16} FGA-002: Mailbox 0</td>
</tr>
<tr>
<td>193</td>
<td>C1_{16}</td>
<td>304_{16} FGA-002: Mailbox 1</td>
</tr>
<tr>
<td>194</td>
<td>C2_{16}</td>
<td>308_{16} FGA-002: Mailbox 2</td>
</tr>
<tr>
<td>195</td>
<td>C3_{16}</td>
<td>30C_{16} FGA-002: Mailbox 3</td>
</tr>
<tr>
<td>196</td>
<td>C4_{16}</td>
<td>310_{16} FGA-002: Mailbox 4</td>
</tr>
<tr>
<td>197</td>
<td>C5_{16}</td>
<td>314_{16} FGA-002: Mailbox 5</td>
</tr>
<tr>
<td>198</td>
<td>C6_{16}</td>
<td>318_{16} FGA-002: Mailbox 6</td>
</tr>
<tr>
<td>199</td>
<td>C7_{16}</td>
<td>31C_{16} FGA-002: Mailbox 7</td>
</tr>
<tr>
<td>200-223</td>
<td>C8_{16} - DF_{16}</td>
<td>320_{16} - 37C_{16} reserved (Unassigned)</td>
</tr>
<tr>
<td>Vector number</td>
<td>Vector address</td>
<td>Assignment</td>
</tr>
<tr>
<td>---------------</td>
<td>---------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>224</td>
<td>E0&lt;sub&gt;16&lt;/sub&gt;</td>
<td>380&lt;sub&gt;16&lt;/sub&gt; FGA-002: Timer</td>
</tr>
<tr>
<td>225</td>
<td>E1&lt;sub&gt;16&lt;/sub&gt;</td>
<td>384&lt;sub&gt;16&lt;/sub&gt; reserved</td>
</tr>
<tr>
<td>226</td>
<td>E2&lt;sub&gt;16&lt;/sub&gt;</td>
<td>388&lt;sub&gt;16&lt;/sub&gt; reserved</td>
</tr>
<tr>
<td>227</td>
<td>E3&lt;sub&gt;16&lt;/sub&gt;</td>
<td>38C&lt;sub&gt;16&lt;/sub&gt; reserved</td>
</tr>
<tr>
<td>228</td>
<td>E4&lt;sub&gt;16&lt;/sub&gt;</td>
<td>390&lt;sub&gt;16&lt;/sub&gt; FGA-002: FMB1 refused</td>
</tr>
<tr>
<td>229</td>
<td>E5&lt;sub&gt;16&lt;/sub&gt;</td>
<td>394&lt;sub&gt;16&lt;/sub&gt; FGA-002: FMB0 refused</td>
</tr>
<tr>
<td>230</td>
<td>E6&lt;sub&gt;16&lt;/sub&gt;</td>
<td>398&lt;sub&gt;16&lt;/sub&gt; FGA-002: FMB1 message</td>
</tr>
<tr>
<td>231</td>
<td>E7&lt;sub&gt;16&lt;/sub&gt;</td>
<td>39C&lt;sub&gt;16&lt;/sub&gt; FGA-002: FMB0 message</td>
</tr>
<tr>
<td>232</td>
<td>E8&lt;sub&gt;16&lt;/sub&gt;</td>
<td>3A0&lt;sub&gt;16&lt;/sub&gt; FGA-002: Abort key</td>
</tr>
<tr>
<td>233</td>
<td>E9&lt;sub&gt;16&lt;/sub&gt;</td>
<td>3A4&lt;sub&gt;16&lt;/sub&gt; FGA-002: ACFAIL</td>
</tr>
<tr>
<td>234</td>
<td>EA&lt;sub&gt;16&lt;/sub&gt;</td>
<td>3A8&lt;sub&gt;16&lt;/sub&gt; FGA-002: SYSFAIL</td>
</tr>
<tr>
<td>235</td>
<td>EB&lt;sub&gt;16&lt;/sub&gt;</td>
<td>3AC&lt;sub&gt;16&lt;/sub&gt; FGA-002: DMA error</td>
</tr>
<tr>
<td>236</td>
<td>EC&lt;sub&gt;16&lt;/sub&gt;</td>
<td>3B0&lt;sub&gt;16&lt;/sub&gt; FGA-002: DMA normal</td>
</tr>
<tr>
<td>237</td>
<td>ED&lt;sub&gt;16&lt;/sub&gt;</td>
<td>3B4&lt;sub&gt;16&lt;/sub&gt; reserved</td>
</tr>
<tr>
<td>238</td>
<td>EE&lt;sub&gt;16&lt;/sub&gt;</td>
<td>3B8&lt;sub&gt;16&lt;/sub&gt; reserved</td>
</tr>
<tr>
<td>239</td>
<td>EF&lt;sub&gt;16&lt;/sub&gt;</td>
<td>3BC&lt;sub&gt;16&lt;/sub&gt; reserved</td>
</tr>
<tr>
<td>240</td>
<td>F0&lt;sub&gt;16&lt;/sub&gt;</td>
<td>3C0&lt;sub&gt;16&lt;/sub&gt; FGA-002: LIRQ0 - watchdog</td>
</tr>
<tr>
<td>241</td>
<td>F1&lt;sub&gt;16&lt;/sub&gt;</td>
<td>3C4&lt;sub&gt;16&lt;/sub&gt; FGA-002: LIRQ1 - FDC 37C65</td>
</tr>
<tr>
<td>242</td>
<td>F2&lt;sub&gt;16&lt;/sub&gt;</td>
<td>3C8&lt;sub&gt;16&lt;/sub&gt; FGA-002: LIRQ2 - CIO1/PC0 (timer 3)</td>
</tr>
<tr>
<td>243</td>
<td>F3&lt;sub&gt;16&lt;/sub&gt;</td>
<td>3CC&lt;sub&gt;16&lt;/sub&gt; FGA-002: LIRQ3 - CIO1/PB0 (timer 2)</td>
</tr>
<tr>
<td>244</td>
<td>F4&lt;sub&gt;16&lt;/sub&gt;</td>
<td>3D0&lt;sub&gt;16&lt;/sub&gt; FGA-002: LIRQ4 - CIO1, CIO2 cascaded</td>
</tr>
<tr>
<td>245</td>
<td>F5&lt;sub&gt;16&lt;/sub&gt;</td>
<td>3D4&lt;sub&gt;16&lt;/sub&gt; FGA-002: LIRQ5 - SCC (Z85C30)</td>
</tr>
<tr>
<td>246</td>
<td>F6&lt;sub&gt;16&lt;/sub&gt;</td>
<td>3D8&lt;sub&gt;16&lt;/sub&gt; FGA-002: LIRQ6 - SCSI (NCR 53C720SE)</td>
</tr>
<tr>
<td>247</td>
<td>F7&lt;sub&gt;16&lt;/sub&gt;</td>
<td>3DC&lt;sub&gt;16&lt;/sub&gt; FGA-002: LIRQ7 - LAN (Am79C965)</td>
</tr>
<tr>
<td>248-254</td>
<td>F8&lt;sub&gt;16&lt;/sub&gt;-FE&lt;sub&gt;16&lt;/sub&gt;</td>
<td>3E0&lt;sub&gt;16&lt;/sub&gt;-3F8&lt;sub&gt;16&lt;/sub&gt; reserved (Unassigned)</td>
</tr>
<tr>
<td>255</td>
<td>FF&lt;sub&gt;16&lt;/sub&gt;</td>
<td>3FC&lt;sub&gt;16&lt;/sub&gt; FGA-002: Empty interrupt</td>
</tr>
</tbody>
</table>
### 7.6 Benchmark Source Code

```
Section 0

opt alt,P=68020,P=68881
xdef .benchex
xdef .BEN1BEG,.BEN1END
xdef .BEN2BEG,.BEN2END
xdef .BEN3BEG,.BEN3END
xdef .BEN4BEG,.BEN4END
xdef .BEN5BEG,.BEN5END
xdef .BEN6BEG,.BEN6END
xdef .BEN7BEG,.BEN7END
xdef .BEN8BEG,.BEN8END
xdef .BEN9BEG,.BEN9END
xdef .BEN10BEG,.BEN10END
xdef .BEN11BEG,.BEN11END
xdef .BEN12BEG,.BEN12END
xdef .BEN13BEG,.BEN13END
xdef .BEN14BEG,.BEN14END

Section 1

Benchmark execution: benchex(address)

movem.l d1-a6,-(a7)
move.l 15*4(a7),a0
jsr (a0)
movem.l (a7)+,d1-a6
rts

Benchmark #1: Decrement Long Word in Memory 10,000,000 Times

lea.l @010(pc),a0
move.l #10000000,(a0)
@020 subq.l #1,(a0)
bne.s @020
rts
@010 ds.l 1

Benchmark #2: Pseudo DMA 1K Bytes 50,000 Times

move.l #50000,d2 ; Do 50000 Transfers
@001 move.w #fff,d3 ; Each is 1K Bytes
lea.l @010(pc),a1 ; A1 Points to Source and Destination
@002 move.l (a1),(a1)+
dbra d3, @002
subq.l #1,d2
bne.s @001
rts
nop
@010 nop
```

PAGE
* * BENCH #3: SUBSTRING CHARACTER SEARCH 100.000 TIMES TAKEN FROM EDN 08/08/85 *
*
MOVE.L #1000000,D4
@002 MOVE.L #15,D0
MOVE.L #120,D1
LEA.L EDN1DAT(PC),A1
LEA.L EDN1DAT1(PC),A0
BSR.S EDN1
SUBQ.L #1,D4
BNE.S @002
RTS
*
****** BEGIN EDN BENCH #1 *******
EDN1 MOVEM.L D3/D4/A2/A3,-(A7)
SUB.W D0,D1
MOVE.W D1,D2
SUBQ.W #2,D0
MOVE.B (A0)+,D3
@010 CMP.B (A1)+,D3
@012 DBEQ D1,@010
BNE.S @090
MOVE.L A0,A2
MOVE.L A1,A3
MOVE.W D0,D4
BMI.S @030
@020 CMP.B (A2)+,(A3)+
DBNE D4,@020
BNE.S @012
@030 SUB.W D1,D2
@032 MOVEM.L (A7)+,D3/D4/A2/A3
RTS
@090 MOVEQ.L #1,-D2
BRA.S @032
****** END EDN BENCH #1 *******
EDN1DAT DC.B '00000000000000000000000000000000'
DC.B '00000000000000000000000000000000'
EDN1DAT1 DC.B 'HERE IS A MATCH00000000000000000000000000000000'
PAGE
*
* BENCH #4: BIT TEST/SET/RESET 100.000 TIMES TAKEN FROM EDN 08/08/85 *
*
MOVE.L #1000000,D4
LEA.L EDN2DAT(PC),A0
@010 MOVEQ.L #1,D0 ; TEST
MOVEQ.L #10,D1
BSR.S EDN2
MOVEQ.L #1,D0
MOVEQ.L #11,D1
BSR.S EDN2
MOVEQ.L #1,D0
MOVE.W #123,D1
BSR.S EDN2
MOVEQ.L #2,D0 ; SET
MOVEQ.L $10, D1
    BSR.S EDN2
    MOVEQ.L $1, D0
    MOVEQ.L $11, D1
    BSR.S EDN2
    MOVEQ.L $1, D0
    MOVE.W $123, D1
    BSR.S EDN2
    MOVEQ.L $3, D0 ; RESET
MOVEQ.L $10, D1
    BSR.S EDN2
    MOVEQ.L $1, D0
    MOVEQ.L $11, D1
    BSR.S EDN2
    MOVEQ.L $1, D0
    MOVE.W $123, D1
    BSR.S EDN2
    SUBQ.L $1, D4
    BNE.S @010
    RTS

* EDN2
    SUB.W $2, D0
    BEQ.S @020
    SUBQ.W $1, D0
    BEQ.S @030

@010
* BFTST (A0){D1:1}
    DC.W $E8D0
    DC.W $0841
    SNE D2
    RTS

@020
* BFSET (A0){D1:1}
    DC.W $EED0
    DC.W $0841
    SNE D2
    RTS

@030
* BFTST (A0){D1:1}
    DC.W $E8D0
    DC.W $0841
    SNE D2
    RTS

EDN2DAT DC.L 0,0,0,0
PAGE

* BENCH #5: BIT MATRIX TRANSPOSITION 100.000 TIMES
* TAKEN FROM EDN 08/08/85
*
MOVE.L #100000,D4
LEA.L EDN3DAT(PC), A0
@002
MOVE.L #7, D0
MOVEQ.L #0, D1
BSR.S EDN3
SUBQ.L $1, D4
BNE.S @002
RTS
Appendix to VMEPROM Benchmark Source Code

* EDN3  MOVEM.L D1-D7,-(A7)
       MOVE.L D1,D2
       MOVE.W D0,D7
       SUBQ.W #2,D7
@010 ADDQ.L #1,D1
       MOVE.L D1,D3
       ADD.L D0,D2
       MOVE.L D2,D4
@020
       BFEXTU (A0){D3:1},D5
       BFEXTU (A0){D4:1},D6
       BFINS D5,(A0){D4:1}
       BFINS D6,(A0){D3:1}
       ADD.L D0,D3
       ADDQ.L #1,D4
       CMP.L D3,D4
       BNE.S @020
       DBRA D7,@010
       MOVEM.L (A7)+,D1-D7
       RTS

EDN3DAT DC.B %01001001
       DC.B %01011100
       DC.B %10001110
       DC.B %10100101
       DC.B %00000001
       DC.B %01110010
       DC.B %10000000
       EVEN
       PAGE

* * BENCH #6: CACHE TEST - 128KB PROGRAM IS EXECUTED 1000 TIMES
* CAUTION: THIS BENCHMARK NEEDS 128 KBYTE MEMORY
*
       LEA.L @010(PC),A2
       MOVE.L #$203A0000,D1 ; OPCODE FOR MOVE.L ($0,PC),D0
       MOVE.L #$20000/4,D2 ; LENGTH IS 128 KBYTE
@004 MOVE.L D1,(A2)+ ; LOAD OPCODE TO MEMORY
       SUBQ.L #1,D2
       BNE.S @004
       MOVE.W #$4E75,(A2) ; APPEND RTS

* PROGRAM IS NOW LOADED -- START 1000 TIMES

MOVE.L #1000,D3
@008 BSR.S @010
       SUBQ.L #1,D3
       BNE.S @008
       RTS

* @010 DC.L 0 ; PROGRAM WILL START HERE
       PAGE

* * BENCH #7: FLOATING POINT 1,000,000 ADDITIONS
*

MOVE.L #1000000,D5
       FMOVE.L #0,FP0
       FMOVE.L #1,FP1
@010 FADD.X FP0,FP1
       SUBQ.L #1,D5
       BNE.S @010
       RTS
* BENCH #8: FLOATING POINT 1.000.000 SINUS

```
MOVE.L #1000000,D5
FMOVE.L #1,FP1
@010 FSIN.X FP1
SUBQ.L #1,D5
BNE.S @010
RTS
```

PAGE

* BENCH #9: FLOATING POINT 1.000.000 MULTIPLICATIONS

```
MOVE.L #1000000,D5
FMOVE.L #1,FP0
FMOVE.L #1,FP1
@010 FMUL.X FP0,FP1
SUBQ.L #1,D5
BNE.S @010
RTS
```

PAGE

* PDOS BENCHMARK #1: CONTEXT SWITCHES

```
MOVE.L #100000,D6
@000 XSWP ;CONTEXT SWITCH
SUBQ.L #1,D6 ;DONE?
BGT.S @000 ;N
RTS
```

PAGE

* PDOS BENCHMARK #2: EVENT SET

```
MOVEQ.L #32,D1 ;SELECT EVENT 32
MOVE.L #100000,D6
```

```
@000 XSEV ;SET EVENT
SUBQ.L #1,D6 ;DONE?
BGT.S @000 ;N
RTS
```

PAGE

* PDOS BENCHMARK #3: CHANGE TASK PRIORITY

```
MOVEQ.L #1,D0 ;SELECT CURRENT TASK
MOVEQ.L #64,D1 ;SET PRIORITY TO 64
MOVE.L #100000,D6
```

```
@000 XSTP ;SET PRIORITY
SUBQ.L #1,D6 ;DONE?
BGT.S @000 ;N
RTS
```
* PDOS BENCHMARK #4: SEND TASK MESSAGE
*
CLR.L D0 ;SELECT TASK #0
LEA.L MES01(FC),A1 ;POINT TO MESSAGE
MOVE.L #100000,D6
*
@000 XSTM ;SEND MESSAGE
XKTM ;READ MESSAGE BACK
SUBQ.L #1,D6 ;DONE?
BGT.S @000 ;N
RTS
MES01 DC.B 'BENCH #13',0
EVEN PAGE

* PDOS BENCHMARK #5: READ TIME OF DAY
*
@000 MOVE.L #100000,D6
@000 EQU *
XRTP
SUBQ.L #1,D6 ;DONE?
BGT.S @000 ;N
RTS
end
7.7 Modifying Special Locations in ROM

Special locations in the VMEPROM binary image define:

- the default setup of the start-up file’s name,
- RAM disk addresses,
- and the user program’s location.

The special locations are defined in the user patch table and can be changed by the user to adapt VMEPROM to the actual working environment.

**IMPORTANT**

Some user patch table entries serve as a group of pre-configured alternatives (e.g., configuration of RAM disk or of program start address). This enables easy configuration selection rather than re-configuring the values. In these cases, the front-panel rotary switches define the actually used configuration. Therefore, the user patch table includes the front-panel rotary switch setting for such cases (see section 6.2.3 “Rotary Switches” on page 128).

Address of user patch table

The address of the user patch table is located at offset \texttt{000C_{16}} relative to the beginning of the VMEPROM image.

**Example:**

Finding the user patch table

```
? M FF00000C L
FF00000C FF00E000 : .

? MD FF00E000 70
FF00E000: 53 59 24 53 54 52 54 00 00 00 00 00 00 00 00 00 SY$STRT............
FF00E010: 00 00 00 00 00 00 00 08 08 00 40 80 00 00 00 08 ..............@
FF00E020: 01 00 FF C0 80 00 00 08 08 00 FC 80 00 00 53 59 ...............SY
FF00E030: 24 44 53 48 00 00 00 00 00 00 00 00 00 00 00 00 $DSK.........
FF00E040: 40 80 00 00 FF C0 80 00 FC 80 00 00 FF 00 DE B6 ................
FF00E050: 55 53 45 52 03 FF 07 FF FF FF FF FF 00 10 00 00 USER............
FF00E060: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................

? _
```
### Table 70: User patch table

<table>
<thead>
<tr>
<th>Offset</th>
<th>Default</th>
<th>Size</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0016</td>
<td>'SYSSTRT',0</td>
<td>DS.B 22</td>
<td>Name of the start-up file. It must be a 0-terminated string.</td>
</tr>
<tr>
<td>1616</td>
<td>8</td>
<td>DS.W 1</td>
<td>1. RAM disk: disk number (MODE 2 rotary switch: bit 1 = 0 and bit 0 = 0)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DS.W 1</td>
<td>Number of 256-byte sectors (2048 = 512 KB)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DS.L 1</td>
<td>Start address in A32 VME space</td>
</tr>
<tr>
<td>2048</td>
<td></td>
<td>DS.W 1</td>
<td>2. RAM disk: disk number (MODE 2 rotary switch: bit 1 = 0 and bit 0 = 1)</td>
</tr>
<tr>
<td>4080.000016</td>
<td></td>
<td>DS.L 1</td>
<td>Number of 256-byte sectors (256 = 64 KB)</td>
</tr>
<tr>
<td>256</td>
<td></td>
<td>DS.W 1</td>
<td>Start address in local SRAM (NVRAM)</td>
</tr>
<tr>
<td>FFC0.800016</td>
<td></td>
<td>DS.L 1</td>
<td>3. RAM disk: disk number (MODE 2 rotary switch: bit 1 = 1 and bit 0 = 0)</td>
</tr>
<tr>
<td>2048</td>
<td></td>
<td>DS.W 1</td>
<td>Number of 256-byte sectors (2048 = 512 KB)</td>
</tr>
<tr>
<td>FC80.000016</td>
<td></td>
<td>DS.L 1</td>
<td>Start address in A24 VME space</td>
</tr>
<tr>
<td>2E16</td>
<td>'SYSDSK',0</td>
<td>DS.B 18</td>
<td>Default name of initialized RAM disk (must be a 0-terminated string).</td>
</tr>
<tr>
<td>4016</td>
<td>4080.000016</td>
<td>DS.L 1</td>
<td>Alternatives for the program start address which is jumped to after kernel initialization. The address actually used can be selected by bit 3 and bit 2 of the MODE 1 rotary switch.</td>
</tr>
<tr>
<td></td>
<td>FFC0.800016</td>
<td>DS.L 1</td>
<td>1. entry: Start program at addr. 4080.000016 (VME).</td>
</tr>
<tr>
<td></td>
<td>FC80.000016</td>
<td>DS.L 1</td>
<td>2. entry: Start program at addr. FFC0.800016 (NVRAM).</td>
</tr>
<tr>
<td></td>
<td>(VMEPROM Shell)</td>
<td>DS.L 1</td>
<td>3. entry: Start program at addr. FC80.000016 (VME).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DS.L 1</td>
<td>4. entry: Start addr. of the VMEPROM shell.</td>
</tr>
<tr>
<td>5016</td>
<td>'USER'</td>
<td>DS.B 4</td>
<td>Disk drivers need this ident to make sure that the data beginning at 5416 is valid.</td>
</tr>
<tr>
<td>5416</td>
<td>0316</td>
<td>DS.B 1</td>
<td>Bit 0 If this bit is 0, no message occurs indicating that VMEPROM is waiting until the hard disk is up to speed. This bit is only considered if bit 1 is set to 1.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Bit 1 If it is 0, VMEPROM does not wait until hard disk is up to speed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Bits 2...7 reserved, should be 0.</td>
</tr>
<tr>
<td>5516</td>
<td>FF16</td>
<td>DS.B 1</td>
<td>reserved</td>
</tr>
<tr>
<td>5616</td>
<td>0716</td>
<td>DS.B 1</td>
<td>SCSI controller ID</td>
</tr>
<tr>
<td>5716</td>
<td>5-times FF16</td>
<td>DS.B 5</td>
<td>reserved</td>
</tr>
<tr>
<td>5C16</td>
<td>16</td>
<td>DS.W 1</td>
<td>Number of 16-Kbyte hashing buffers (to improve disk access speed). Valid entries are numbers from 1 to 32.</td>
</tr>
<tr>
<td>5E16</td>
<td>0000.000016</td>
<td>DS.L 1</td>
<td>reserved</td>
</tr>
</tbody>
</table>
Reprogramming the flash memory

The following procedure describes how to reprogram the on-board system flash memory.

**IMPORTANT**

If there is another CPU or memory board available on the VMEbus, it should be used to save the current content of the system flash into a file:

1. Copy the binary image to the local RAM of the other CPU board via the VMEbus:

   \[ \text{BM } FF000000, FF400000, \text{destination} \]

2. Save the binary image into a file.

After saving the current content of the on-board system flash it can be reprogrammed:

1. Enter the FGA-002 boot software by simultaneously asserting the reset and abort switch and then releasing the reset switch.

2. Initialize the FGA-002 and make the main memory available:

   \[ \text{INIT} \]

3. Check whether switch SW10-3 is set appropriately to enable writing to the system flash memory: OFF = writing enabled.

4. Copy the VMEPROM binary image of the system flash into RAM:

   \[ \text{BM } FF000000, FF040000, 0 \]

5. Modify the code of the VMEPROM image in RAM.

6. Erase the page in system flash memory where VMEPROM is stored (these are the first 256 Kbytes):

   \[ \text{FERASE } SYS\_FLASH, 0, 40000 \]

7. Reprogram the flash memory:

   \[ \text{FPROG } SYS\_FLASH, 0, 0, 40000 \]

8. Reboot the system to test the changes.
7.8 Binding Applications to VMEPROM

In general, there are two ways to bind an application program in the flash memory to the VMEPROM kernel. The first way keeps the original flash memory contents unchanged and uses external memory, the second needs to reprogram the system flash. In all cases the application program is executed in user mode. The \texttt{XSUP} system call can be used to switch to supervisor mode.

7.8.1 Using External Memory

The application can be put into an external RR-2 or RR-3 EPROM board on the VMEbus. In this case, the front panel switches of the CPU board must be set so that the application program is started after VMEPROM is booted. In this instance the user stack is located at the top of the tasking memory and the supervisor stack is located within the task control block. The supervisor stack has a size of 500 bytes. Registers are not predefined. If the reserved supervisor stack space is not sufficient, the stack pointer has to be set to point to an appropriate address in RAM.

7.8.2 Using System Flash Memory

Since the VMEPROM image needs about 512 Kbytes of the system flash memory, there are still 3.5 Mbytes of memory available. These can be used to hold a user’s application.

For the reprogramming of the flash memory, see section 7.7 “Modifying Special Locations in ROM” on page 174.

Binding the Application

1. Enter the boot software and copy the system flash memory contents into RAM.
2. Merge your own application with the VMEPROM code in RAM.
3. Alter the necessary entries in the VMEPROM binary image. Be sure to use the correct addresses. They must be calculated for system flash, not for RAM! Depending on the time the application should be called, this will be the
   - ‘Pointer to VMEPROM Initialization’ (early exit),
   - one of the 4 entries at offset 4016 in the user patch table (see page 175),
   - or ‘Pointer to VMEPROM Shell’ (late exit, replacement of the shell). In this case, the application will be called with the address of the TCB and SYRAM on the stack: 4(A7) Long word containing the start address of the TCB and 8(A7) Long word containing the start address of the system RAM (SYRAM).
A C-program at this address could look like this:

```c
main (struct TCB *pTCB, struct SYRAM *pSYRAM)
{ }
```

For further information, see table 63 “Layout of system flash memory” on page 133. For information on programming the modified image into system flash see section 7.7 “Modifying Special Locations in ROM” on page 174.
8 FGA Boot

The booter on this CPU board is the FGA-002 boot software, also called FGA Boot. It provides

- the initialization of the board’s hardware,
- debugger commands,
- and utility functions.

8.1 Boot Sequence

At first FGA Boot initializes the devices on the board and checks if the board is System Controller (slot-1). If so, it turns on the user LED (UL) and enables the FGA-002 arbiter. For more details about the slot-1 functionality, please refer to “VMEbus Slot-1” on page 86.

Then the firmware in the second boot flash (at address FFE8.0000\text{16}) or the system flash memory (at address FF00.0000\text{16}) is started. It is also possible to specify another module address that will be stored in the battery-buffered SRAM (see SETUP command).

**IMPORTANT**

The binary images on these locations must be program modules, i.e. they must provide an SSP (stack pointer) at offset 0\text{16} and a PC (program counter) at offset 4\text{16}.

No modules found

If no program modules are found, the debugger will be started instead.

Hardware initialization

During hardware initialization the following steps are performed:

- Leave boot mode of the FGA-002 and the RIALTO bus bridge (map boot PROM from 0000.0000\text{16} to FFE0.0000\text{16}).

- Setup the VMEbus A32 slave window (A24 slave window as well if enabled, see section 8.2.12 “SETUP – Change Initialization Values” on page 193). The snooping window size is also set. Per default, the VMEbus window is configured to enable VMEbus accesses to the entire memory.

- For the SCSI device the EA and the FA bit in the DCNTL register are set to 1.

- Relocate the LAN device to FFF0.0000\text{16}. After hardware initialization the LAN device still is in 16-bit mode.

- Enable FGA-002 arbiter if the CPU board is system controller.
Any program starting after FGA Boot has to initialize the CPU registers (e.g., PCR, CACR) anew as there is no default setting for them when FGA Boot exits.

### Figure 10  Boot up procedure

<table>
<thead>
<tr>
<th>7-seg. display</th>
<th>Actions / State</th>
</tr>
</thead>
<tbody>
<tr>
<td>undef.</td>
<td>The Boot ROM (at address FFE0.0000&lt;sub&gt;16&lt;/sub&gt;) is mapped to 0000.0000&lt;sub&gt;16&lt;/sub&gt;. The CPU loads its initial stack pointer (SSP) and initial program counter (PC) from locations 0&lt;sub&gt;16&lt;/sub&gt; and 4&lt;sub&gt;16&lt;/sub&gt;.</td>
</tr>
<tr>
<td>undef.</td>
<td>Install exception handler and execute C startup code.</td>
</tr>
<tr>
<td>undef.</td>
<td>Check reset condition and leave FGA-002’s boot mode. Now the Boot ROM resides at address FFE0.0000&lt;sub&gt;16&lt;/sub&gt;. Check if Abort Switch is asserted ➞ abort.</td>
</tr>
<tr>
<td>off</td>
<td>Read board ID from port and initialize 7-Segment Display.</td>
</tr>
<tr>
<td>0</td>
<td>Initialize CPU registers CACR, ITT&lt;sub&gt;x&lt;/sub&gt; and DTT&lt;sub&gt;x&lt;/sub&gt;, disable all caches.</td>
</tr>
<tr>
<td>1</td>
<td>Initialize the front-panel serial I/O port 1.</td>
</tr>
<tr>
<td>2</td>
<td>Initialize the CIO devices.</td>
</tr>
<tr>
<td>3</td>
<td>Identify board features (interfaces) and read serial ID-ROM.</td>
</tr>
<tr>
<td>4</td>
<td>Determine the processor’s clock frequency (with cache enabled).</td>
</tr>
<tr>
<td>5</td>
<td>Determine capacity of main memory.</td>
</tr>
<tr>
<td>6</td>
<td>Calculate checksum of SRAM parameters (register and system values).</td>
</tr>
<tr>
<td></td>
<td>is it correct ?</td>
</tr>
<tr>
<td></td>
<td>Y</td>
</tr>
<tr>
<td></td>
<td>N</td>
</tr>
<tr>
<td></td>
<td>% Set SRAM parameters to default values.</td>
</tr>
<tr>
<td>7</td>
<td>Check if the board is system controller via bit 0 of the “Slot-1 Status Register”. This bit can be set by the slot-1 autodetection or SW6-1 and SW6-2.</td>
</tr>
<tr>
<td></td>
<td>bit 0 == 0 ?</td>
</tr>
<tr>
<td></td>
<td>N</td>
</tr>
<tr>
<td></td>
<td>Y</td>
</tr>
<tr>
<td></td>
<td>% Board is System Controller, set bit 2 (ARBITER) in CTL1 value.</td>
</tr>
<tr>
<td>8</td>
<td>Test for EAGLE modules (not applicable for the SYS68K/CPU-60).</td>
</tr>
<tr>
<td>9</td>
<td>Read front-panel rotary switches and store to SRAM.</td>
</tr>
</tbody>
</table>
**Figure 10**  
Boot up procedure (continued)

<table>
<thead>
<tr>
<th>7-seg. display</th>
<th>Actions / State</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Check for Firmware Module to start:</td>
</tr>
<tr>
<td>A</td>
<td>is variable Start Module at Address even ?</td>
</tr>
<tr>
<td></td>
<td>(see Note 1)</td>
</tr>
<tr>
<td>Y</td>
<td>is firmwarebase defined ?</td>
</tr>
<tr>
<td></td>
<td>(see Note 2)</td>
</tr>
<tr>
<td>%</td>
<td>second Boot ROM ?</td>
</tr>
<tr>
<td></td>
<td>firmwaremodule = variable</td>
</tr>
<tr>
<td>firmwaremodule</td>
<td>firmwaremodule = firmwarebase</td>
</tr>
<tr>
<td>firmwaremodule</td>
<td>firmwaremodule = BOOT_ROM2</td>
</tr>
<tr>
<td>firmwaremodule</td>
<td>firmwaremodule = SystemFlash</td>
</tr>
<tr>
<td></td>
<td>Set startModule=OK (assume firmwaremodule is executable).</td>
</tr>
</tbody>
</table>

While (1):

- abort || (startModule==ERROR) ?
  - N
  - abort && new slot# ?
    - N
    - Store new slot number (rotaries) and halt.
  - Y
    - Print FORCE-Boot banner and start debugger
      - SHELL (type EXIT to leave the shell)

- Initialize FGA-002, arbiter, User LED and other hardware. Set up VMEbus A32 slave window (and A24 if enabled).
  - is it a Power-On Reset ?
    - N
    - Y
      - Clear DRAM (fill with 0) to initialize parity.

- Call user program (address at offset C16 of the Boot ROM).
  - is firmwaremodule executable ?
    - Y
      - Start firmwaremodule (exit boot software)........
    - N
      - startModule=ERROR

**Note 1:** This variable can be set with the `SETUP s` command.
**Note 2:** This is an entry at offset 2C16 of the Boot ROM which can be patched by the user.
8.2 Debugger Commands

Automatic start
If no program modules are found during the boot sequence, the debugger will automatically be started.

Manual start
To start the debugger manually, both rotary switches must be set to F16 and the abort switch must be kept asserted while reset. Note that in this case the hardware needs to be initialized by the INIT command.

IMPORTANT
Bus errors
When accessing the DRAM from the debugger after a power up, this may cause bus errors due to uninitialized parity.

• Use the DRAMINIT command to initialize the DRAM (see “DRAMINIT – Initialize DRAM” on page 185).

Debugger commands
The common debugger commands are described in the FORCE Gate Array FGA-002 User’s Manual. Additional commands available for this version of the booter are described in this section.

Quick overview
To get a short description of all commands enter:

```
FORCE-BOOT> ? <cr>
```

Line Editor
The shell knows the following control characters for line editing:

ESC or CONTROL C Break current command line.
CONTROL A Recall previous command.
CONTROL B Go to begin of line.
CONTROL E Go to end of line.
CONTROL H Move cursor one character left.
CONTROL L Move cursor one character right.
CONTROL D Delete character under cursor.
DEL Delete character left from cursor.
CONTROL \ Delete from cursor until end of line.
CONTROL O Delete whole line.
CONTROL I Toggle between insert/overwrite mode.
ENTER or RETURN Execute command line.

8.2.1 AS – Line Assembler

Format

```
AS address
```

The AS command invokes the line assembler of FGA Boot. It can assemble and disassemble all 68020/30/40 mnemonics. When the AS command is invoked, it displays the current address and disassembles the opcode at this location.
After the prompt on the next line, the user can enter one of the following:

1. A valid 680x0 mnemonic. Some addressing modes allow omission of arguments. These addressing modes can be entered by omitting the argument and typing the dividing character ",".

Examples:

   CLR.W ([$1, A0], D0, W, $2)
   CLR.W ([$1, A0], , $2)
   CLR.W ([, A0], ,)

2. A ’#’ sign followed by the new address changes the address counter to this absolute address.

3. An ’=’ disassembles the same location again.

4. A ’+’ or <return> disassembles the next location.

5. A ’+’ or ’-’ sign followed by the number of bytes increases/decreases the address counter.

6. A ’.’ or <ESC> allows to exit the line assembler and returns control to the command interpreter.

7. <Ctrl-A> copies the current disassembled opcode in the line buffer. This allows editing the current mnemonic.

All immediate values, addresses, and offsets used within mnemonics are assumed to be entered in decimal. Hex values have to be specified by a dollar sign ‘$’. In addition, binary values can be entered by a preceding percent sign ‘%’, octal values by an at sign ‘@’. The disassembler displays all values in hex representation. The line assembler accepts also pseudo opcodes of the form DC.B, DC.W, and DC.L to define constant data storage. An ASCII pattern can be stored by using DC.B with the format ’DC.B "text"’. All characters after the ’”’ will be interpreted as ASCII characters and stored in memory.

The disassembler displays all illegal or unknown opcodes as DC.W.

Example:

```
FORCE-BOOT> AS 8000
$00008000 : ORI.B #0, D0
      : MOVE.L #$123, D1
$00008006 : ORI.B #0, D0
      : -6 move 6 bytes back
$00008000 : MOVE.L #$123, D1
      : <Ctrl-A> recall line
$00008006 : ORI.B #0, D0
      : ADDI.L #20, D1
$0000800c : ORI.B #0, D0
      : . leave assembler
FORCE-BOOT> _
```
8.2.2 BANNER – Display Banner Again

Format

BANNER

The BANNER command displays the same information as is displayed when starting the debugger. This is useful to get the current settings after modifying values via the SLOT or VMEADDR command.

8.2.3 CONT – Continue with Calling Routine

Format

CONT

The CONT command allows to leave the debugger after it was entered from a user’s application via BSR (entry address stored at FFE0.003016) or via an exception (for setting a vector, use the address stored at FFE0.003416).
All registers will be restored before leaving the debugger via an RTS or RTE instruction.

Example:

```
FORCE-BOOT> CONT
```

8.2.4 DI – Disassembler

Format

DI address
DI address, count

The DI command causes the disassembler to be invoked and displays the mnemonic, starting at the specified address. If the count parameter is given, the specified number of lines (mnemonics) will be displayed. If count is omitted, a full page is displayed on the terminal and the user is prompted to continue disassembly (enter <Return>) or to abort (enter any other key).
The disassembler supports all 68020/30/40 mnemonics.
Example:

```
8000 5
00008000 MOVE.L #$123,D1
00008006 ADDI.L #$14,D1
0000800c ORI.B #0,D0
00008010 ORI.B #0,D0
00008014 ORI.B #0,D0
```

8.2.5 DRAMINIT – Initialize DRAM

**Format**

DRAMINIT

The DRAMINIT command will only have an effect if called after power up for the first time. It fills the complete DRAM with 0 if dynamic RAM with parity is used on the board. This forces the parity bits to be correct and prevents parity errors when reading from memory locations that have not been written previously. If there is SRAM on the board, it will not be initialized.

Example:

```
FORCE-BOOT> DRAMINIT
FORCE-BOOT> _
```

8.2.6 FERASE – Erase Flash Memories

**Format**

FERASE flashbank
FERASE flashbank,flashoffset,length

The FERASE command allows to erase flash memory banks.

Format 1 of the command erases the whole flash memory bank.

Format 2 allows to specify a region to erase.

**IMPORTANT**

This region must exactly match the page boundaries of the flash devices. Example: If the SYS_FLASH bank consists of four 28F008 (1 M * 8 bit) devices in parallel with a page size of 64 Kbyte each, the minimum size of one erasable region is 256 Kbyte (64 KB * 4).
Parameters

`flashbank`

Symbolic name or base address of the flash memory bank that should be erased. The following symbolic names are currently supported:

- `BOOT_FLASH` (first) boot flash
- `BOOT_FLASH1` first boot flash
- `BOOT_FLASH2` second boot flash
- `SYS_FLASH` system flash
- `USER_FLASH` user flash

`flashoffset`

Optional relative byte offset within the flash bank.

`length`

Optional length in bytes. If `flashoffset` and `length` are not specified, the whole bank will be erased.

Example:

```
FORCE-BOOT> FERASE
Usage: FERASE <flashbank>,[<flashoffset>,<length>]
Parameter <flashbank> is the base address of the flash bank
or one of the following defines:
  BOOT_FLASH1 BOOT_FLASH2 SYS_FLASH1 USER_FLASH1

FORCE-BOOT> FERASE BOOT_FLASH1
Do not reprogram BOOT_FLASH1, this would destroy the booter
Device is write protected

FORCE-BOOT> FERASE SYS_FLASH 80000 40000
Erasing flash memory ... done.
FORCE-BOOT> _
```

8.2.7 FPROG – Program Flash Memories

Format

```
FPROG flashbank,source
FPROG flashbank,source,flashoffset
FPROG flashbank,source,flashoffset,length
```

The `FPROG` command allows to program flash memory banks.

Format 1 of the command programs the whole flash memory bank with the data stored at the specified source address.

Format 2 additionally allows to specify a destination offset within the flash memory bank and programs all remaining space (from offset to end of flash bank).
Format 3 of the command also specifies the number of bytes to program.

Parameters

**flashbank**

Symbolic name or base address of the flash memory bank that should be programmed. The following symbolic names are currently supported:

- **BOOT_FLASH** (first) boot flash
- **BOOT_FLASH1** first boot flash
- **BOOT_FLASH2** second boot flash
- **SYS_FLASH** system flash
- **USER_FLASH** user flash

**flashoffset**

Optional relative byte offset within the flash bank. If no offset is specified, 0 is assumed.

**length**

Optional length in bytes. If no length is specified, all remaining space of the flash bank will be programmed.

Example:

```
FORCE-BOOT> FPROG

Usage: FPROG <flashbank>,<source>[,<flashoffset>[,<length>]]
       Parameter <flashbank> is the base address of the flash bank
       or one of the following defines:
       BOOT_FLASH1  BOOT_FLASH2  SYS_FLASH1  USER_FLASH1

FORCE-BOOT> FPROG BOOT_FLASH1,100000
Do not reprogram BOOT_FLASH1, this would destroy the booter
Device is write protected

FORCE-BOOT> FPROG BOOT_FLASH2,100000
Programming flash memory

  0 |##################################################| 100%
Done.

FORCE-BOOT> _
```
8.2.8 GO – Go to Subroutine

**Format**

```
GO address
```

The GO command calls a subroutine at the specified address. To get back into the debugger an RTS instruction must be executed by the subroutine.

**Example:**

```
FORCE-BOOT> AS 0
$00000000 : ORI.B #0,D0
: RTS
$00000002 : ORI.B #0,D0
: .
FORCE-BOOT> GO 0
FORCE-BOOT> _
```

8.2.9 LO – Load S-Records to Memory

**Format**

```
LO [host commands]
LO offset[,host commands]
LO V[,host commands]
LO offset,V[,host commands]
LO E
```

The LO command allows to load S-records from the console port into memory and to verify the memory contents.

The optional parameter *host commands* allows to specify a list of commands that will be sent to the host to initiate the data transfer, e.g. `cat testfile`.

Format 1 of the command is a standard download. Data will be loaded to the absolute addresses as specified by the S-records.

Format 2 contains a parameter *offset* that specifies the value that is added to the absolute addresses of the S-records. This allows to modify the storage address while downloading.

Format 3 and 4 are the same as previously described, except that no data will be loaded to memory, but a comparison takes place between the memory contents and the S-record data. This allows to verify the data.

Format 5 displays the number of errors that occurred during the last download.
Example:

The following program originally located at address 0010.0000\textsubscript{16} should be loaded to 0010.0200\textsubscript{16} via a ‘tip’ connection.

Original program:

```
00100000 MOVE.L #$123456,D0
00100006 NOP
00100008 SUBQ.L #1,D0
0010000a BNE.B $100006
0010000c RTS
```

S-record file test.x:

```
S000000FC
S212100000203C001234564E71538066FA4E7530
S804000000FB
```

```
FORCE-BOOT> LO 200 add offset 200\textsubscript{16} to addresses
use ‘~C’ to execute local command
~CLocal command? cat test.x use ‘cat test.x’ to transfer file
away for 2 seconds!

FORCE-BOOT> DI 100200 5 list program
00100200 MOVE.L #$123456,D0
00100206 NOP
00100208 SUBQ.L #1,D0
0010020a BNE.B $100206
0010020c RTS
FORCE-BOOT> _
```
8.2.10 NETLOAD – Load File via Network to Memory

Format

```
NETLOAD  filename, start_address
[, ethernet_number]
[, target_IP#, server_IP#]
```

The NETLOAD command loads the specified binary file via Ethernet to memory. It uses the TFTP (Trivial File Transfer Protocol) to connect the CPU-60 to the server where the file is located. Therefore, a TFTP server must exist. In detail:

- **RARP**:  
  - If no IP (Internet Protocol) numbers are specified (**target_IP#** and **server_IP#**), NETLOAD sends a RARP packet (Reverse Address Resolution Protocol) to translate the board’s Ethernet number into an IP address. A RARP server and a translation table are required for this task. E.g. on a UNIX system the file `/etc/ethers` must contain the board’s Ethernet number.
  
  - After the board has received its own IP number, a TFTP request is sent to this server which has replied the RARP. The server now starts sending the requested file. On a UNIX system, the file must be located in the `/tftpboot` directory.

- **ARP**:  
  - When the board’s own target IP and server IP number are given by **target_IP** and **server_IP#**, a standard ARP request is broadcasted to get the Ethernet number of the server.
  
  - After the board has received the ARP reply, a TFTP request is sent to the specified server. This server now starts sending the requested file. On a UNIX system, the file must be located in the `/tftpboot` directory.

Parameters

- **filename**
  Name of the file to load from the host to the local memory.

- **start_address**
  Local start address where the contents of the file should be stored to. Note, that the NETLOAD command on the CPU-60 uses space 0000.0000\(^16\) to 0000.2FFF\(^16\) of the main memory for data buffers. This region cannot be used for other purposes during the transfer.

- **ethernet_number**
  The Ethernet number that should be used for the CPU board for the TFTP transfer. It must be specified as 6 two-digit hex numbers separated by colons. If the Ethernet number is stored on the board, this parameter is optional.
**target_IP#**

IP (Internet Protocol) number of the CPU board

**server_IP#**

IP (Internet Protocol) number of the server where the file is located

**Example:**

File *test* is located in the `/tftpboot` directory of a UNIX system. It contains the text *This is a test*. The configuration file `/etc/ethers` has an entry with the board’s Ethernet number and the name of the board (its IP address), for example:

```
0:80:42:3:88:88 board1
```

---

**8.2.11 NETSAVE – Save Data via Network to File**

**NETLOAD**  

```
filename,start_address,end_address
[,ethernet_number]
[, target_IP#, server_IP#]
```

**Format**

```
NETSAVE filename,start_address[,ethernet_number]
```

The **NETSAVE** command saves the specified memory region into a file located in a server system. Similar to **NETLOAD** this is done via the TFTP protocol.

**IMPORTANT**

This file cannot be created. It must already exist with correct write permissions!

**Parameters**

`filename`

Name of the file to save data from the local memory into. This filename must already exist on the host.
**start address**

Local start address of the region that should be saved into the file. Note, that the NETLOAD command on the CPU-60 uses space 0000.0000₁₆ to 0000.2FFF₁₆ of the main memory for data buffers. This region cannot be used for other purposes during the transfer.

**end address**

Local end address of the region that should be saved into the file.

**ethernet number**

The ethernet number that should be used for the CPU-60 board for the TFTP transfer. It must be specified as 6 two-digit hex numbers separated by colons. If the ethernet number is stored on the board, this parameter is optional.

**target_IP#**

IP (Internet Protocol) number of the CPU board

**server_IP#**

IP (Internet Protocol) number of the server where the file is located

**Example:**

The following commands save the memory region 0010.0100₁₆ to 0010.011F₁₆ into the file *test*. The file’s content will be overwritten.

```plaintext
FORCE-BOOT> BF 100100 100120 "# Another Test #" P
FORCE-BOOT> MD 100100 20
00100100:  23 20 41 6e 6f 74 68 65  72 20 54 68 74 20 23 # Another Test #  20 23 # Another Test #
00100110:  23 20 41 6e 6f 74 68 65  72 20 54 68 74 20 23 # Another Test #
FORCE-BOOT> NETSAVE test 100100 10011F 00:80:42:03:88:88
LAN-controller at address FEF80000 set to Ethernet 00:80:42:03:88:88
Transmitting RARP-REQUEST...
LAN-controller at address FEF80000 set to Ethernet 00:80:42:03:88:88
Transmitting RARP-REQUEST...
Reception of RARP-REPLY
Transmitting TFTP-REQUEST...
PACKET:1 - saved $00100100..$0010011F (32 bytes)
```

FORCE-BOOT> _
8.2.12 SETUP – Change Initialization Values

Format

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>SETUP F or SETUP</td>
</tr>
<tr>
<td>SETUP S</td>
</tr>
</tbody>
</table>

The **SETUP** command is used to change SRAM parameters. **SETUP F** and **SETUP** allow to modify the initialization values of the FGA-002 as described in the *FGA-002 User’s Manual*. **SETUP S** allows to set up additional system and application values which are also stored in the battery-buffered SRAM. These values can be read by an application software via the utility interface.

For further information, please see section 8.3 “FGA Boot Utility Functions” on page 196.

After modifying any entries, the **INIT** command must be executed to recalculate the SRAM checksum and to validate the new values.

The following entries exist:

<table>
<thead>
<tr>
<th>Name</th>
<th>Default value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start Module at Address</td>
<td>FFFF.FFFF&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
</tbody>
</table>

The user can specify the address of a program module here. A module must provide a SSP (stack pointer) at offset 0<sub>16</sub> and a PC (program counter) at offset 4<sub>16</sub>. If a value of FFFF.FFFF<sub>16</sub> is set (default), this entry will be ignored and FGA Boot starts the firmware in the second boot ROM or the system flash memory. If the entry contains a valid module base address, the specified program will be executed instead.

**Hint:** If the debugger shell should be started, this entry must be set to an address where there is an invalid module, e.g. FF7F.FFF0<sub>16</sub>.

<table>
<thead>
<tr>
<th>Name</th>
<th>Default value</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Flags</td>
<td>0000&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
</tbody>
</table>

This entry is board specific. On the CPU-60 it is defined as following:

- bit 15..10: reserved and must be set to 0
- bit 9..8: VMEbus timer, if the CPU board is system controller; bit 9 and 8 are stored in VMEBUSTIMER[1..0] of the memory configuration register (see table 26 “MCR, memory configuration register” on page 54)
- bit 7..1: reserved and must be set to 0
- bit 0: Enable (1) or disable (0) VMEbus A24 slave window

<table>
<thead>
<tr>
<th>Name</th>
<th>Default value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Flags</td>
<td>0000&lt;sub&gt;16&lt;/sub&gt;</td>
</tr>
</tbody>
</table>

This entry is reserved for an application and will not be used by FGA Boot. A call to utility function 39 allows to read it.
8.2.13 SLOT – Change Slot Number and VMEbus Slave Address

The *SLOT* command allows to modify the VMEbus slave address, the VMEbus address of the mailbox array (register MYVMEPAGE of the FGA-002), and the FMB slot number (register FMBCTL) as follows:

\[
\begin{align*}
\text{VMEbus slave address (A32)} &= 8000.0000_{16} \\
&\quad + 0400.0000_{16} \times (\text{slot#} - 1) \\
\text{VMEbus slave address (A24)} &= 00.0000_{16} \\
\text{Mailbox base address (A16)} &= 8000_{16} + 0400_{16} \times (\text{slot#} - 1) \\
\text{FBM slot number} &= \text{slot#}
\end{align*}
\]

The size of the VMEbus slave window will be set as large as the local memory size of the board. This is exactly the same as setting the rotary switches to the corresponding slot number and asserting the abort key while reset. Note, that the A24 slave window must be enabled separately via *SETUP S*.

**IMPORTANT**

After setting a new slot number the *INIT* command must be executed to recalculate the SRAM checksum and to validate the new values. The *BANNER* command may be used to display the current settings.
Example:

```
FORCE-BOOT> SLOT 5
Use the INIT command to recalculate the SRAM checksum!
FORCE-BOOT> INIT
FORCE-BOOT> 
```

### 8.2.14 VMEADDR – Change VMEbus Slave Address

**Format**

```
VMEADDR slave address
VMEADDR slave address,slave window
```

The `VMEADDR` command allows modifying the VMEbus slave address without modifying any other settings.

- VMEbus slave address (A32) = `slave address`
- VMEbus slave address (A24) = `slave address A23..0`

By means of the first format of the command the size of the VMEbus slave window is set as large as the local memory size of the board. The second format allows to specify the size of the VMEbus slave window manually.

**IMPORTANT**

After executing this command the **INIT** command must be executed to recalculate the SRAM checksum and to update the FGA-002 registers and the snooping window size in the RIALTO bus bridge. The **BANNER** command may be used to display the current settings.

**Example:**

The following example sets the VMEbus A32 slave address of the board to `8320.0000_16` and the window size to 1 Mbyte. It can now be accessed from `8320.0000_16` to `832F.FFFF_16`. The A24 slave window will be set to `20.0000_16` and is also 1 Mbyte large. Note, that the A24 slave window must be enabled separately via **SETUP S**.

```
FORCE-BOOT> VMEADDR 83200000,100000
Use the INIT command to recalculate the SRAM checksum!
FORCE-BOOT> INIT
FORCE-BOOT> 
```
8.3 FGA Boot Utility Functions

FGA Boot provides utility functions which can be called from user applications.

**Common utility functions**
The common utility functions are described in the *FORCE Gate Array FGA-002 User’s Manual.*

The following additional utility functions available for this booter version are described in this section:

- Extended flash memory programming
- Erase flash memories
- Get system values in SRAM
- Get application values in SRAM
- Get Ethernet number
- Get memory limits

**C-calling conventions**
The interface expects C-like calling conventions.

**IMPORTANT**

- The utility interface must be called in supervisor mode.
- It does not install its own stack but will use the application’s stack.
- All parameters must be placed as long values on the supervisor stack.
- The first parameter must be pushed onto the stack as the last one. It must include the function number of the requested function.

### Table 71 Stack frame

<table>
<thead>
<tr>
<th>Offset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A7</td>
<td>return address</td>
</tr>
<tr>
<td>+4</td>
<td>function number</td>
</tr>
<tr>
<td>+8</td>
<td>first argument of function</td>
</tr>
<tr>
<td>+12</td>
<td>second argument of function</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Calling a utility function

Calling sequence

1. Retrieve the entry address from location BootROM+000816.
2. Push the parameters (32 bit values) onto the stack. Remember to push the function number at last.
3. Call the FGA-Boot utility interface in supervisor mode via JSR.
4. Retrieve the return code from the interface (register D0).
5. Clean up the stack (arguments are still on the stack).

Calling from C

Example:

Calling the utility function fctNo from a C-programming language source.

```c
#define BootROM 0xFFE00000

main(
    long (*util)(long fctNo, ...);
    long ret;
    util = *((long **) (BootROM + 0x0008));
    ret = util (.....);
)
```

8.3.1 Extended Flash Memory Programming

This routine allows partial programming of flash memories. The FGA-002’s timer is used for timing during execution of the routine.

Function number 36 (=2416)

Syntax long util (36, flashbank, source, offset, length)

Parameters

flashbank
Base address of the flash memory bank that is to be programmed.

source
Source address of the data to program.

offset
Relative byte offset within the flash bank.

length
Length in bytes. If length is 0, all remaining space of the flash bank will be programmed.
Returns

0  OK       – OK, no errors
1  CLEAR_ERROR – flash device cannot be set to 0 for erasing
2  INVAL_PARMS – invalid parameters
3  ERASE_ERROR – flash device is not erasable
4  WRITE_ERROR – programming error
5  ILL_WIDTH   – illegal flash bank width was detected
6  UNKNOWN_ID  – unknown flash device identifier
7  CAPACITY    – device is too small
8  WRITEPROTECT – flash bank is write protected
9  NO_VPP      – no programming voltage
10 SELECT_ERROR – cannot select the specified flash bank
11 UNIMP_CMD    – unimplemented command
12 UNSUP_DEV    – unsupported flash device type

8.3.2 Erase Flash Memories

The function allows partial erasing of flash memory banks if the devices support page erasing mode. The FGA-002’s timer is used for timing during execution of the routine.

Function number 37 (=25_{16})

Syntax

long util (37, flashbank, offset, length)

Parameters

flashbank

Base address of the flash memory bank that should be erased.

offset

Relative byte offset within the flash bank (see below).

length

Length in bytes (see below).

IMPORTANT offset and length must exactly match the page boundaries of the flash devices. For example: If the system flash bank consists of four 28F008 (1 M * 8 bit) devices in parallel with a page size of 64 Kbyte each, the minimum size of one erasable region is 256 Kbyte (64 KB * 4). If offset and length are both set to 0, the whole flash bank is erased.
Returns

0  OK    – OK, no errors
1  CLEAR_ERROR – flash device cannot be set to 0 for erasing
2  INVAL_PARMS – invalid parameters
3  ERASE_ERROR – flash device is not erasable
4  WRITE_ERROR – programming error
5  ILL_WIDTH – illegal flash bank width was detected
6  UNKNOWN_ID – unknown flash device identifier
7  CAPACITY – device is too small
8  WRITEPROTECT – flash bank is write protected
9  NO_VPP – no programming voltage
10 SELECT_ERROR – cannot select the specified flash bank
11 UNIMP_CMD – unimplemented command
12 UNSUP_DEV – unsupported flash device type

8.3.3  Get System Values in SRAM

This function sets a pointer to the base address of the system values stored in the SRAM. It also returns the size of this structure (# of bytes).

Function number  38 (=2616)

Syntax  

long util (38, SYS_VALUES **pSysValues)

Parameter  

pSysValues  

Address (!) of a pointer to structure SYS_VALUES. This pointer will be set by the routine to that location where the system values start in SRAM.

typedef packed struct
{  
   unsigned long  startModule;
   unsigned short  sysFlags;
}  SYS_VALUES;

Returns  Size of structure in bytes
### 8.3.4 Get Application Values in SRAM

This function sets a pointer to the base address of the application values stored in the SRAM. It also returns the size of this structure (# of bytes).

**Function number** 39 (\(=27_{16}\))

**Syntax**

```
long util (39, APPL_VALUES **pApplValues)
```

**Parameter**

- `pApplValues`
  
  Address (!) of a pointer to structure `APPL_VALUES`. This pointer will be set by the routine to that location where the application values start in SRAM.

```c
typedef packed struct
{
    unsigned short applFlags;
    unsigned long applValue;
} APPL_VALUES;
```

**Returns**

Size of structure in bytes

### 8.3.5 Get Ethernet Number

This function copies the board’s Ethernet number (6 bytes) to the specified buffer. The return value includes the status of this operation.

**Function number** 40 (\(=28_{16}\))

**Syntax**

```
long util (40, long intfNumb, char *pEtherAdr)
```

**Parameters**

- `intfNumb`
  
  Interface number, must be set to 0.

- `pEtherAdr`
  
  Pointer to buffer where the Ethernet number should be stored into.

**Returns**

- 0 OK
- (-1) ERROR, no Ethernet number available.
8.3.6 Get Memory Limits

This function determines the start address and the total size of the shared RAM (DRAM + user SRAM) and sets the specified variables to the respective values. The return value includes the status of this operation.

Function number 41 (=29₁₆)

Syntax

```
long util (41, long *pBaseVar, char *pSizeVar)
```

Parameters

- `pBaseVar`
  Address (!) of the variable where the start address of the shared RAM is stored into. For the SYS68K/CPU-60 this variable is always 0 because the shared RAM starts at 0.

- `pSizeVar`
  Address (!) of the variable where the total size (in Byte) of the shared RAM is stored into.

Returns

- 0 OK
- (-1) ERROR, cannot determine size of shared RAM.