Special Registers and Exception Handling on MPC875

Z. Jerry Shi
Department of Computer Science and Engineering
University of Connecticut

CSE4903: Microprocessor Laboratory
MP875 block diagram

Courtesy of Freescale Semiconductor
Memory map

OS data and exception handler

Data and code in DRAM

16 KB memory resource decided by the upper 16 bits of Internal Memory Map Register (IMMR)

Other purposes
Five General Purpose I/O Ports

- Ports A, B, C, D, E
- Port B is shared with the parallel interface port (PIP) and other functions such as SMC, IDMA, etc.
- Port B has 18 bits
  - PB14 to PB31
- All signals can be open-drain
- If the bit in PBPAR is cleared: general-purpose I/O
- If the bit in PBPAR is set: dedicated on-chip peripheral signals
- Four control registers
Port B Pin Assignment Register (PBPAR)

- DDn = 0 : General-purpose I/O.
- DDn = 1 : Dedicated peripheral function.

<table>
<thead>
<tr>
<th>Field</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reset</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>R/W</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R/W</td>
</tr>
<tr>
<td>Addr</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0xABC</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field</th>
<th>DD16</th>
<th>DD17</th>
<th>DD18</th>
<th>DD19</th>
<th>DD20</th>
<th>DD21</th>
<th>DD22</th>
<th>DD23</th>
<th>DD24</th>
<th>DD25</th>
<th>DD26</th>
<th>DD27</th>
<th>DD28</th>
<th>DD29</th>
<th>DD30</th>
<th>DD31</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reset</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
</tr>
<tr>
<td>Addr</td>
<td>0xABE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Port B Pin Assignment

<table>
<thead>
<tr>
<th>Signal</th>
<th>Signal Function</th>
<th>Input to On-chip Peripherals (Default)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PB31</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PR30</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PB29</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PB28</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PB27</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Signal Function**

<table>
<thead>
<tr>
<th>Signal</th>
<th>PBPAR[DDn] = 0</th>
<th>PBDIR[DRn] = 0</th>
<th>PBDIR[DRn] = 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>PB31</td>
<td>UT = 1</td>
<td>UT = 0</td>
<td>UT = 1</td>
</tr>
<tr>
<td>Port B31</td>
<td>MII TXCLK</td>
<td>RMII1-REFCLK</td>
<td>SPISEL</td>
</tr>
<tr>
<td>PR30</td>
<td>UT = 1</td>
<td>UT = 0</td>
<td>UT = 1</td>
</tr>
<tr>
<td>Port B30</td>
<td>—</td>
<td>SPICI K</td>
<td>SPICI K = GND</td>
</tr>
<tr>
<td>PB29</td>
<td>UT = 1</td>
<td>UT = 0</td>
<td>UT = 1</td>
</tr>
<tr>
<td>Port B29</td>
<td>—</td>
<td>SPIMOSI</td>
<td>SPIMOSI = V&lt;sub&gt;DD&lt;/sub&gt;</td>
</tr>
<tr>
<td>PB28</td>
<td>UT = 1</td>
<td>UT = 0</td>
<td>UT = 1</td>
</tr>
<tr>
<td>Port B28</td>
<td>BRGO4</td>
<td>SPIMISO</td>
<td>SPIMISO = SPIMOSI</td>
</tr>
<tr>
<td>PB27</td>
<td>UT = 1</td>
<td>UT = 0</td>
<td>UT = 1</td>
</tr>
<tr>
<td>Port B27</td>
<td>BRGO1</td>
<td>I2CSDA</td>
<td>I2CSDA = V&lt;sub&gt;DD&lt;/sub&gt;</td>
</tr>
</tbody>
</table>
Port B Data Direction Register (PBDIR)

- **DRn = 0**: Select the signal for general-purpose input or select peripheral function 0
- **DRn = 1**: Select the signal for general-purpose output or select peripheral function 1

<table>
<thead>
<tr>
<th>Field</th>
<th>Reset</th>
<th>R/W</th>
<th>Addr</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>0xAB8</td>
</tr>
<tr>
<td>0</td>
<td>—</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>—</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>DR14</td>
<td>DR15</td>
<td>R/W R/W</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field</th>
<th>DR16</th>
<th>DR17</th>
<th>DR18</th>
<th>DR19</th>
<th>DR20</th>
<th>DR21</th>
<th>DR22</th>
<th>DR23</th>
<th>DR24</th>
<th>DR25</th>
<th>DR26</th>
<th>DR27</th>
<th>DR28</th>
<th>DR29</th>
<th>DR30</th>
<th>DR31</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reset</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
</tr>
<tr>
<td>Addr</td>
<td>0xBA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Port B Open-Drain Register (PBODR)

- Indicates when the port signals are configured
- Bits 14 and 15 of PBODR are not implemented.
- PBODR is cleared by system reset.
- $OD_n = 0$: the I/O signal is actively driven as an output
- $OD_n = 1$: open-drain driver. Actively driven low. Otherwise, it is three-stated
Port B Data Register (PBDAT)

- Port B data register.

<table>
<thead>
<tr>
<th>Bits</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>D14</td>
<td>D15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reset</td>
<td>Undefined</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Addr</td>
<td>0xAC4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bits</th>
<th>16</th>
<th>17</th>
<th>18</th>
<th>19</th>
<th>20</th>
<th>21</th>
<th>22</th>
<th>23</th>
<th>24</th>
<th>25</th>
<th>26</th>
<th>27</th>
<th>28</th>
<th>29</th>
<th>30</th>
<th>31</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field</td>
<td>D16</td>
<td>D17</td>
<td>D18</td>
<td>D19</td>
<td>D20</td>
<td>D21</td>
<td>D22</td>
<td>D23</td>
<td>D24</td>
<td>D25</td>
<td>D26</td>
<td>D27</td>
<td>D28</td>
<td>D29</td>
<td>D30</td>
<td>D31</td>
</tr>
<tr>
<td>Reset</td>
<td>Undefined</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td></td>
</tr>
<tr>
<td>Addr</td>
<td>0xAC6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Pins 29 and 30 of Port B

• Control debug LED
• Output 0: LED is on
• Steps:
  – Set proper registers so pins 29 and 30 are configured as general-purpose output pins
  – Clear bits to turn on LEDs
    • Do not disturb other bits
MPC875 interrupt structure
Decrementer (DEC)

- A 32 bit register decremented periodically
- An exception is raised when bit 0 change from 0 to 1
  - From 0x00000000 to 0xFFFFFFFF
  - Controlled by software
- Can be accessed with `mtspr` and `mfspr`
  ```
  mtspr 22, R1  mtdec R1
  mfspr R1, 22  mfdec R1
  ```
- The exception handler is located at offset 0x00900
  - Depends on the interrupt prefix (IP) bit in MSR
    - 0x0000-0900 or 0xFFF0-0900
    - More on Machine Status Register (MSR) in a moment
- Keyed register
  - You have to unlock the key before writing to DEC
Timebase Status and Control Register (TBSCR)

A 16-bit register located at

\[(\text{IMMR} \& \ 0xFFFF0000) + 0x200\]

Bit 15 (TBE)

= 0: Disables timebase and decremener operation
= 1: Enables timebase and decremener operation

Keyed register !!
Enable timebase and decremener

- TBSCR and DEC are keyed registers
  - TBSCRK: (IMMR & 0xFFFF0000) + 0x300
  - TBK: (IMMR & 0xFFFF0000) + 0x30C

Write any other value to the key register

Write 0x55CCAA33 to the key register
Steps to enable decremeneter exception

• Set proper exception handler at 0x0000_0900
  – Directive to put code at certain location
  – Explicitly copy code to 0x900

• Enable timebase
  – Unlock TBSCR: write 0x55CCAA33 to TBSCRK
  – Set bit 15 in TBSCR

• Set decremeneter
  – Unlock DEC first: write 0x55CCAA33 to TBK

• Enable external interrupt
  – Set bit 16 (EE) in MSR
Exception handler

- Take the exception
- Save registers
- Switch the Status of LEDs
- Restore registers
- Return from the exception handler
Supervisor-level registers

- Machine State Register (MSR)
- Machine Status Save/Restore Register
  - Used to save machine status on exceptions
  - SRR0: points to an instruction that all prior instructions have completed
  - SRR1: keeps a copy of important bits in MSR
- SPRG0 to SPRG3
  - Four special registers for fast state save
- Decrementser (DEC)
- Many others
Machine State Register (MSR)

- **Bit 16 (EE): External interrupt enable**
  - 1 : enabled

- **Bit 17 (PR): Privilege level**
  - 1 : user mode

- **Bit 25 (IP): Interrupt prefix**
  - Decide the prefix of exception vector offset
    - 0x000n-nnnn or 0xFFFn-nnnn

More on PE: page 2-21

Use “rds msr” to check the register in the debugger
Machine Status Save/Restore Register

rfi:
Go to SRR0
MSR ← SRR1

SRR0: Next instruction
SRR1: Bits in MSR
MSR is set accordingly
Exception classifications

- Asynchronous or synchronous?
  - Asynchronous: by events external to the execution
  - Synchronous: by instructions

- Asynchronous:
  - Non-maskable
  - Maskable

- Synchronous:
  - Precise
  - Imprecise
Precise exceptions

- Instruction dispatching is halted
- All prior instructions has reported the exceptions they will cause
- Instructions that precede the operation complete execution in the context in which they were initiated
Exception priorities

1. Non-maskable asynchronous
   • System reset
   • Machine check
2. Synchronous precise
3. Synchronous imprecise
4. Maskable asynchronous
   • External interrupt
   • Decrementer
Recoverability after an exception

- Cannot always recover from system reset and machine check
- All other exceptions may be restartable
  - Make sure SRR0 and SRR1 are saved early
- MSR[RI] notifies the exception handler whether the processor is in a restartable state
  - It is saved in SSR1[RI] after exception
  - It is cleared on the entry of an exception handler
  - It is restored from SSR1[RI] when exception handler returns
- Set MSR[RI] when it is safe to take an exception
  - Clear MSR[RI] when saving or restoring SSR0 and SSR1
- In critical section where MSR[EE] is not set, MSR[RI] should remain set if SRR0 and SRR1 are safe
  - If an exception occurs, the process is restartable
- MSR[RI] is not always checked, especially when handling exceptions that are restartable by convention
Nested exceptions

1. Any exception.
2. RFI
3. Reset, NMI, machine check, debug(2), and synchronous. If MSR.EE = 1, then external and decrementer.
4. RFI. User checks SRR1.RI = 1.
Recoverable interrupts

- Possible to take Path 2 after 1, 3, and 4
  - Save SRR0 and SRR1 before allowing exceptions
  - Check MSR[RI] in ESR2 and do not use rfi if not recoverable
- Two registers to change EE and RI bits fast
  ```
  mtspr EIE, 0
  mtspr NRI, 0
  ```
  - EIE: External interrupt enable
    - End of handler’s prologue, enable nested external interrupts;
    - End of critical code segment in which external interrupts were disabled
  - EID: External interrupt disable, but other exception are recoverable
    - End of handler’s prologue, keep external nested interrupts disabled;
    - Start of critical code segment in which external interrupts are disabled
  - NRI: Nonrecoverable interrupt (start of handler’s epilogue)

<table>
<thead>
<tr>
<th>SPR</th>
<th>Mnemonic</th>
<th>MSR[EE]</th>
<th>MSR[RI]</th>
</tr>
</thead>
<tbody>
<tr>
<td>80</td>
<td>EIE</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>81</td>
<td>EID</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>82</td>
<td>NRI</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Readings

- PE: Chapter 2: Register set
  - Specially the exception handling registers listed on page 2-19
- PE: Chapter 6: Exceptions
- RM: Chapter 2: Memory map
- RM: Chapter 6: Exceptions
- RM: Chapter 10: System interface unit
Issues when using MPC875 board

- Emulated system calls like printf may interfere with I/O access and exception
- USB/TAP may not be able to handle USB 2.0 data rate
  - From Freescale support: This is a USB 2.0 issue that occurs when the port runs at its highest speed and overruns the probe interface. To change the USB port to a slower speed, go into Control Panel > System > Hardware > Device Manager > Universal Serial Bus controllers. Now disable all instances (two in all) of Standard Enhanced PCI to USB Host Controller. The USB interface should run at the slower USB 1.1 speeds and eliminate the problem.