Lab 1 - Using TS-7250 I/O

Week 1 - Introduction to TS-7250 board

ECE 4220/7220
Real Time Embedded Computing
Spring 2016
University of Missouri
Introduction

• Lab Guidelines
• TS-7250 Board Setup
• Program Development with Eclipse
ARM Processor

• On the ARM processor there are seven ports which can control up to 8 pins:

  Port sizes are as follows:
  Port A - 8 bits
  Port B - 8 bits
  Port C - 1 bits
  Port E - 2 bits
  Port F - 3 bits
  Port G - 2 bits
  Port H - 4 bits.

• Each port has a data register and a data direction register.

• Register addresses are aligned on word (32-bit) boundaries.
Example: Port A Registers

- **PADR**: Values read will reflect the external value of the pin while values written will change the value of the pin.

- **PADDR**: A value of 1 will configure the corresponding pin to output while a value of zero sets to input.
## Location of Registers

<table>
<thead>
<tr>
<th>Address</th>
<th>Read Location</th>
<th>Type</th>
<th>Write Location</th>
<th>Reset Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x8084_0000</td>
<td>PADR</td>
<td>R/W</td>
<td>PADR</td>
<td>Note 1</td>
</tr>
<tr>
<td>0x8084_0004</td>
<td>PBDR</td>
<td>R/W</td>
<td>PBDR</td>
<td>Note 1</td>
</tr>
<tr>
<td>0x8084_0008</td>
<td>PCDR</td>
<td>R/W</td>
<td>PCDR</td>
<td>Note 1</td>
</tr>
<tr>
<td>0x8084_000C</td>
<td>Reserved</td>
<td>-</td>
<td>Reserved</td>
<td>-</td>
</tr>
<tr>
<td>0x8084_0010</td>
<td>PADDR</td>
<td>R/W</td>
<td>PADDR</td>
<td>0x00</td>
</tr>
<tr>
<td>0x8084_0014</td>
<td>PBDDR</td>
<td>R/W</td>
<td>PBDDR</td>
<td>0x00</td>
</tr>
<tr>
<td>0x8084_0018</td>
<td>PCDDDR</td>
<td>R/W</td>
<td>PCDDDR</td>
<td>0</td>
</tr>
<tr>
<td>0x8084_001C</td>
<td>Reserved</td>
<td>-</td>
<td>Reserved</td>
<td>-</td>
</tr>
<tr>
<td>0x8084_0020</td>
<td>PEDR</td>
<td>R/W</td>
<td>PEDR</td>
<td>Note 2</td>
</tr>
<tr>
<td>0x8084_0024</td>
<td>PEDDDR</td>
<td>R/W</td>
<td>PEDDDR</td>
<td>0x03</td>
</tr>
<tr>
<td>0x8084_0028</td>
<td>Reserved</td>
<td>-</td>
<td>Reserved</td>
<td>-</td>
</tr>
<tr>
<td>0x8084_002C</td>
<td>Reserved</td>
<td>-</td>
<td>Reserved</td>
<td>-</td>
</tr>
<tr>
<td>0x8084_0030</td>
<td>PFDR</td>
<td>R/W</td>
<td>PFDR</td>
<td>Note 1</td>
</tr>
<tr>
<td>0x8084_0034</td>
<td>PFDDR</td>
<td>R/W</td>
<td>PFDDR</td>
<td>0x00</td>
</tr>
<tr>
<td>0x8084_0038</td>
<td>PGDR</td>
<td>R/W</td>
<td>PGDR</td>
<td>Note 1</td>
</tr>
<tr>
<td>0x8084_003C</td>
<td>PGDDR</td>
<td>R/W</td>
<td>PGDDR</td>
<td>0x0C</td>
</tr>
<tr>
<td>0x8084_0040</td>
<td>PHDR</td>
<td>R/W</td>
<td>PHDR</td>
<td>Note 1</td>
</tr>
<tr>
<td>0x8084_0044</td>
<td>PHDDR</td>
<td>R/W</td>
<td>PHDDR</td>
<td>0x00</td>
</tr>
</tbody>
</table>
The TS-7250 Board

• The TS-7250 board uses some of these I/O pins for itself, but leaves 20 pins available for user development.
• However, the addresses in the previous table are physical addresses which a user process does not have access to by default.
• In Linux, a user’s C code can obtain access by using the mmap() system call on the /dev/mem special file to map the areas of physical address space into process user address space.

(How to open /dev/mem ?)
Using mmap

```c
void *mmap(void *addr, size_t len, int prot, int flags, int fields, off_t off);
```

- `addr`: where in the user address space to place the mapped area → 0
- `len`: Number of bytes to map → will be rounded to a page
- `prot`: Allows the read/write access
- `flags`: determines how changes to the map affect other processes
- `fields`: file descriptor to the special file → must open the file first using open()
- `off`: offset in the special file that the mapping will start → must be a page multiple

- `mmap` returns a pointer to the beginning of the mapped area.
- Pointers to registers can be offsets of the returned pointer.
- **Example**: `ptr = (unsigned long *)mmap(NULL, getpagesize(), ... ...PROT_READ | PROT_WRITE, MAP_SHARED, fd, 0x80840000);`

(How can we offset `ptr` to point to the required registers?)
Auxiliary Board

• Many of the labs this semester will use the Auxiliary board which connects to the DIO1 header on the TS-7250 board.

• The DIO1 header connects to the pins controlled by PortB and one pin from Port F.

Warning:

• Other DIO1 Port functionality, used for dedicated TS-7250 functions, utilize these same control registers. All accesses to these registers should use read-modify-write cycles.
Auxiliary Board: Schematic

Notice that the least significant bit of the registers is on the right! (e.g. PBDR: B7 B6 B5 B4 B3 B2 B1 B0).
Be careful when assigning values to the ports.
Bit Masking

• Many times we will want to change one I/O pin.
• This is difficult since we must write to all bits of the port.
• Use **bit masking** to change the state of the pin and leave all others unchanged.
• Bitwise operations: AND, OR, XOR.
• These can be used to set, clear, and toggle pins.
Example

• We want to turn on the red LED of the Auxiliary board.
• Our pointer to PBDR is named pbdr.
• The bit of port B associated with the red LED is bit 5.
• To turn on the light we want to set the bit to 1
  \[ \text{*pbdr} = \text{*pbdr} \mid 0x20; \text{ or } \text{*pbdr} \mid= 0x20; \]
Where 0x20 = 0010 0000 b

\[ \uparrow \quad \uparrow \quad \uparrow \]

Bit 7   Bit 5   Bit 0