Reading Assignment

- Software and Hardware Engineering (new version): Chapter 11
  
  Or
  
  - SHE (old version): Chapter 7
  
  And
  
  - M68HC12 Data Sheet: Chapter 1 (1.6.8 - end), Chapter 5 and 6

EEL 4744C: Microprocessor Applications

Lecture 6

Part 2

MC68HC12 Parallel I/O

Operating Modes (1)

- Selected by BKGD, MODA, MODB pins when RESET* is applied (total 8, special modes allow greater access to protected control registers)
- Normal single-chip mode
  - All I/O and memory contained within device
    - HC12B32: 32KB EEPROM, 1KB RAM
  - Self-contained except for clock source and reset circuit
  - Pros: external interfaces do not have to be designed
  - Cons: RAM capacity is quite limited and may not fit large applications

Programmer’s I/O Model

- All I/O and control of I/O is performed using 512 control registers
  
  - Initially mapped to addresses $0000$-$01FF$ (but may be changed via writing to special register)
  
  - Except for a few exceptions, most I/O ports require initialization before use via programming bits in the control registers
HC12 Parallel I/O Ports

- I/O ports can be accessed just like memory (e.g. memory mapped I/O, can use various addressing modes)

  REGS: EQU $0000; Register Base address
  PORTB: EQU $01 ; Offset to PORTB

  ; Read Port B
  ldaa PORTB ; Direct Addressing
  lda #REGS
  ldaa PORTB, x ; Indexed
  lda VECTOR
  lda [0,x] ; Indexed-Indirect
  VECTOR: DC.W PORTB ; Address of PORTB

HC12B32 Parallel I/O Ports

- Ports A and B used for I/O in single-chip mode, or multiplexed external address and data bus pins in expanded mode

  - Port E used as 6 I/O lines plus IRQ* and XIRQ* interrupt pins in single-chip mode; control signals in expanded modes (more later)
  - Port S used as 8 I/O lines (default), or one SCI (2 lines), one SPI (4 lines), and 2 I/O lines (more later)
  - Port T used as 8 I/O lines (default) or 8 timer bits (more later)
  - Port AD may be 8-bit input port (default) or used for 8 analog inputs to on-chip A/D converter (more later)

Data Direction Registers

- Used to control input or output use of bits in a port (1=output, 0=input)

- When CPU reset, all registers (except a few such as in Port E) placed in input mode

- If port a mixture of input and output bits:
  - Writing to port affects only those set as outputs
  - Reading from port returns values on input bits as well as last value(s) on output bits

HC12 Parallel I/O Ports

- Ports C, D, F, G, H, J are not available on HC12B32 (they may appear in the examples in the text book or lecture)
  - Read SHE (old version) pp 169-179 to obtain basic information

- We will learn ports E, S, T, and AD as well as the corresponding control bits in great detail later
### Data Direction Registers Example

```
PORTD: EQU 5 ; PORTD address
DDRD: EQU 7 ; Data Direction Reg
OBITS: EQU %11110000 ; Bits to be output

bset DDRD, OBITS ; Set direction register

ldaa #%11110000
staa PORTD
```

### Pull-up Control and Reduced Drive

- Good design practice to tie unused input pins to either logic-1 or logic-0
- HC12 provides Pull-Up Control Register (PUCR) to enable pull-up resistors on any ports configured as inputs
  - on reset, all pull-up resistors are enabled
- RDRIV (Reduced Drive of I/O Lines): reduce the drive level of output pins for power saving and lower RFI (disabled on reset)

### Writing HC12 I/O Software

- Initialization: set up the function of the ports and the direction of data flow
- Data input/output: read/write data
- Synchronization:
  - Real-time synchronization (e.g. delay loop)
  - Polled I/O
  - Interrupt (more later)

### Polled I/O Example
**Polled I/O Assembly Code**

PORTJ EQU $28 : Port J address
PORTH EQU $24 : Port H address
DDRH EQU $25 : Port H data direction
BIT1 EQU %00000001
O_BITS EQU %00001111

- Initialization
- Set up PORTH[3:0] to be output
- Output data to Port H
- Wait until the status bit, Port J, Bit-0 is 1

```
staa PORTJ,BIT0,SPIN1
```

```
PORTH
```

Dr. Tao Li

**Hardware Handshaking I/O**

- No dedicated h/w for this
- Use general-purpose I/O bits plus s/w similar to polling

- Unlike polling, handshaking goes both ways

**Hardware Handshaking I/O Example**

(a) Handshake Output

(b) Handshake Input

**Handshaking I/O Assembly Code**

```
lea PORTJ,BIT0
```

Dr. Tao Li

```
lea PORTH
```

Dr. Tao Li

Dr. Tao Li
Handshaking I/O Assembly Code

```assembly
; Handshaking input data from Port E
01A AC3898
28 ; Set READY_FOR_DATA on Port J, bit-2
01b 472864F0
37 ; Wait until the status bit, Port J, bit-2 is 1
01c 472864F0
37 ; Read the data
023 4624
28 ; Ack
024 TAD0306
37 ; Write the R/W* and E bus
027 4D0506
27 ; Write the ADDR/Data
02A 38 data; D0 1
02B 39 data; D0 1
```

HC12B32 Expanded Modes

- Expanded mode provides address, data, and control buses at the expense of Ports A, B and some pins in Port E.

Memory Mapping in HC12B32 Expanded Modes

- Normal expanded-narrow & normal expanded-wide modes
  - In both forms of expanded mode, some I/O ports are used for creating a system bus.
  - On HC12B32
    - Ports A and B serve as pins for address bus (ADDRn..0)
    - Ports A (and B if wide mode) as pins for multiplexed data bus
    - Ports E as pins for control bus (e.g. R/W*, E clk, LSTRB*)
**Accessing External Memory and Ports in Expanded (Wide) Modes**

- In expanded-wide mode, the B32 has a multiplexed 16-bit address and data bus
  - With a 16-bit address bus, the B32 can access $2^{16} = 65,536$ bytes of data
  - With a 16-bit data bus, the B32 can access 16 bits (two bytes) in a single bus cycle
- In expanded mode, the B32 uses Port A and Port B as the multiplexed address/data bus

- Timing is controlled by the E clock
  - When the E clock is low, the B32 places the address on the multiplexed bus
    - Port A is used for address bits 15-8
    - Port B is used for address bits 7-0
  - When the E clock is high, the B32 uses the multiplexed bus for data bus
    - Port A is used for D15-D8 [data for even (high) byte]
    - Port B is used for D7-D0 [data for odd (low) byte]

---

For example, if accessing the sixteen-bit word at address 0x4000 (the bytes at addresses 0x4000 and 0x4001):
- Port A will access the byte at address 0x4000
- Port B will access the byte at address 0x4001

**Note:** The follow example is for illustration purpose. On HC12B32 expanded modes, external memory should be mapped between $8000 - FFFF$
To determine whether it should access one byte or two bytes, the B32 uses the LSTRB* and A0 lines:

- LSTRB* low: accessing the lower (odd) byte of a word
- LSTRB* high: accessing the upper (even) byte of a word
- A0 low: accessing the upper (even) byte of a word
- A0 high: accessing the lower (odd) byte of a word

Accessing External Memory and Ports in Expanded (Wide) Modes

<table>
<thead>
<tr>
<th>LSTRB</th>
<th>A0</th>
<th>Type of Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Aligned 32-bit access of an even address</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>8-bit access of an even address</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>8-bit access of an odd address</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Misaligned Not aligned on external bus</td>
</tr>
</tbody>
</table>

- **ldaa $4000**
  - Address on bus: A0: 0 LSTRB*: 1
- **ldaa $4001**
  - Address on bus: A0: 1 LSTRB*: 0
- **ldd $4000**
  - Address on bus: A0: 0 LSTRB*: 0

A Simple Parallel Input Port

- Create an input port at address 0x4000 (an even address, or high byte)

A Simple Parallel Output Port

- Create an output port at address 0x4001 (an odd address, or low byte)
An Output Port which can be Read

- Create an output port which can be read at address 0x4001 (an odd address, or low byte)

A Parallel Input/Output Port

- Create a input/output port at address 0x4001 (an odd address, or low byte)

Expanded Mode Memory

Expanded-Narrow Mode Memory
Expanded-Wide Mode Memory

32k x 1 RAM (addresses $0000 - $7FFF), 16-bit data bus, for Expanded Wide Mode

Address Bus (ADDR15 to ADDR14)
- A0 - A15

Data Bus (DATA0 - DATA15)
- 00 - D7
- N'W
- 'WE

N'W
- 'OE

Dr. Tao Li