Lecture 19
Midterm Review

Instructor

Prof. Chengmo Yang

Office: 201C Evans Hall

Hours: M 2:30—3:30PM

Email: chengmo@udel.edu

Teaching Assistants

	Bo Lu	Chen Liu	Brian Lucas
Office:	Evans 302	Evans 154	Evans 121
Hours:	Th 1:30-3pm	Wed 2:30-4pm	**Tu 1:45-3:15pm**
Email:	lubo@udel.edu	liuchen@udel.edu	lucasb@udel.edu

Midterm Information

- Midterm this Friday
  - In class, close book, close notes, no calculator…
- Number of questions?
  - 5-6
- Topics?
  - Lectures
  - Projects

Lecture Topics

Topic	Lecture #
State Machine Review	3
C Review	4
Embedded System Intro	6
C to MIPS, Tool Chain	8
MIPS ISA, Assembly	12, 14-15
Interrupts	16-17
Buses	18-19

Lecture Topics

- State Machine
  - State diagram, flow chart, Mealy v.s Moore
- C code
  - Variables, Operators, Conditions, Statements…
  - Compiler, assembler, linker, load
  - Program memory layout
  - Procedure call steps
- Embedded Systems
  - Characteristics, technology trends

Lecture Topics

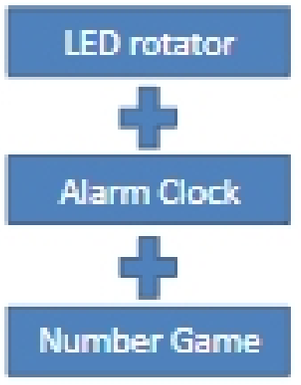
- MPIS ISA
  - RISC vs. CISC
  - 3 types of operands
  - Registers vs. memory
  - Memory organization
    - Byte vs. word, Big vs. little endian
  - 3 instruction formats
    - Translate assembly to binary and vise versa
  - 5 instruction classes
    - Translate C to MIPS and vise versa

Lecture Topics

- Interrupts
  - Interrupts vs. polling
  - Multiple, nested interrupts
  - Configure (CN) interrupts for our board
- Buses
  - Bus principles
  - Low speed bus: SPI, I2C, UART
  - Arm AMBA: AHB, APB
  - Bus on our boards
  - Pipelining

Project Topics

- Code state machines in C
  - Next state logic, output logic…
- Configure the board
  - Input/output ports, LAT vs. PORT
  - Interrupts…
- Monitor inputs
  - Debouncing
  - Push buttons, keypads
- Drive outputs
  - On board LEDs
  - LED Pmod
  - 7-segment display



Example Question 1

Below are some statements and procedures used in project2 for configuring interrupts.

```
1.    CNEN    = 0x003C;
2.    CNPUE   = 0x003C;
3.    CNCON   = 0x8000;
4.    IPC6SET = 0x00140000;
5.    IEC1SET = 0x0001;
6.    IFS1CLR = 0x0001;
7.    INTEnableInterrupts();
8.    INTDisableInterrupts();
9.    INTEnableSystemSingleVectoredInt();
10.   INTEnableSystemMultiVectoredInt();
```

Add comments to each statement describing its function.

Question 1 – Solution

Add comments to each statement describing its function.

```
CNEN    = 0x003C; // Enable individual CN pins CN2, CN3. CN4, and CN5;
CNPUE   = 0x003C;// Enable weak pull ups for pins  CN2, CN3. CN4, and CN5;
CNCON   = 0x8000;// Enable Change Notice module
IPC6SET = 0x00140000;// Set priority level=5
IEC1SET = 0x0001;// Enable Change Notice interrupts
IFS1CLR = 0x0001;// Clear the interrupt flag status bit
INTEnableInterrupts(); // Enable interrupts
INTDisableInterrupts(); // Disable interrupts
INTEnableSystemSingleVectoredInt();//This function enables system wide
                                   single-vector interrupt handling.
INTEnableSystemMultiVectoredInt();//This function enables system wide
                                  multi-vector interrupt handling.
```

Example Question 2

The following code shows a procedure *showNumber()* that displays two 7-segment hex digits through a single selection pin and 7 data pins. To display a digit, the procedure calls *displayDigit*, with the number stored in *display_value*.

```
#define REFRESH_RATE 250
void displayDigit(unsigned char sel /*select signal*/,
        unsigned char num /*number to display*/);
void showNumber(){
        int i;
        displayDigit(0, number[display_value&0xF] );
        for (i=0; i<REFRESH_RATE; i++);
        displayDigit(1, number[(display_value>>4)&0xF]);
            for (i=0; i<REFRESH_RATE; i++);
}
```

a) Rewrite *showNumber()* to display two decimal digits instead of hex.

b) Suppose now we have n (a unknown constant) digits on a single Pmod. The data pins are still 7, while the selection pins have been increased. Rewrite *showNumber()*so that it can display n hex digits (instead of 2) through these pins. Write your code in a way that no manner how large n is, the code is still scalable and readable.

Question 2 – Solution

(a) Rewrite *showNumber()* so that it can display two decimal digits instead of hex. Assume the number to be displayed is still stored in *display_value*.

```
void showNumber(){
        int i;
        displayDigit(0, number[display_value%10] );
        for (i=0; i<REFRESH_RATE; i++);
        displayDigit(1, number[(display_value/10)%10]);
        for (i=0; i<REFRESH_RATE; i++);
}
```

Example: *display_value=34.*

display_value%10=4,

display_value/10%10=3.