Microprocessors & Interfacing

Review

Lecturer : Annie Guo

Lecture Overview

- What have we learned/gained in this course?
- What can we do and where can we go after completing this course?
- About final exam

What have we learned/gained?

- Basics of microprocessor systems
  - What it is about?
- AVR and AVR programming
  - How to program the processor to solve different problems?
- Interfacing
  - How to make microprocessors communicate with external components
    - Take inputs and output results
- Microcontroller application development
  - Put together – develop a microcontroller application

Basics of Microprocessors

- Microprocessors are digital systems
  - Can be programmed to perform varied functions on different information.
- Information can be encoded
  - Numbers in binary code, 2's complement code, …
  - Characters in ASII code
  - Keys on the keypad + key operations for different input information
  - Etc.

Basics of Microprocessors

- Five basic components of hardware computer systems
  - Datapath, control unit, memory, input/output devices
- Microprocessor
  - Datapath + control unit → on one chip
- Microcontroller
  - Microprocessor+memory+ peripherals → on one chip
- Computer application systems
  - Hardware + software
- ISA
  - Interface between hardware and software

ISA

- Instruction set architecture
  - Things that assembly programmers can use and should know
  - Specifications that hardware designers should follow and implement*

ISA (cont.)

- Instruction set architecture
  - Including
    - Instruction set
    - Supported native data type
    - Registers
      - What kinds of registers are available?
      - How to use them?
    - Memory models
      - How are instructions or data stored?
      - How can memory be accessed?
    - Interrupts schemes
      - What kinds of interrupts are available?
      - How to use them?

AVR and AVR Programming

- Program structures
  - Instructions, directives and comments
- Data and data structure implementation
  - Constants, variables
  - Integers, characters, arrays, …
  - Bits
    - Control bits
- Basic programming
  - Sequential operations, if-then-else, loops
- Function implementation
  - Stack and stack frames
  - Functions and function calls

Interfacing

- I/O devices
  - Input devices: switch, push button, keypad
  - Output devices: LED bar, LCD
- Basic interaction approaches:
  - Polling and Interrupts
- Basic communication types
  - Parallel and serial input/output
- Handling analog input/outputs
  - ADC, DAC, PWM

What can we do?

- Enjoy good existing designs
  - Know what, how and why
- Maintain (existing) microprocessor application systems
  - Debug problem
  - Enhance system
    - For more features
    - For performance
    - For upgrade to different version/system
- Design a microprocessor application system
  - For an alternative and efficient solution.
    - For example, software locker
- Be able to proceed to extended/advanced areas for breath/in-depth study

Where can we go?

- Proceed to advanced computing courses
  - Computer architecture
    - How to improve the processor and computer hardware system?
  - Compiler
    - For large software development, automatic machine code generation
  - Operating system
    - How to make good and efficient use of the computer hardware system
  - Networking
    - How to handle complicated communications and make communication efficient?
  - Embedded system design
    - How to develop an efficient processor for a specific application
    - How to design and implement an embedded system with microcontrollers, microprocessors and other specific designed hardware components.

About Final Exam

- Duration: 2 hours
- Close book exam
  - Instruction set and ASCII table provided
  - Materials of weeks 1-12
- Three sections
  - Multiple choices
  - Small questions
  - Medium-fairly large questions

About Final Exam (cont.)

- Things will not be tested
  - Predefined labels for I/O registers
    - Register names
    - Bit names and the bit values for setting a peripheral
  - Using comment lines for any information you need to know related to an **I/O setting**. For example,

```
; insert code here to modify the control register of Timer0 to
  set up the timer for 1 second duration
; insert code here to enable timer0 overflow interrupt
```

Examples: multiple choices

1. If the decimal number (-11) is stored in an 8-bit register, the binary contents of the register should be

   (a) -00000011

   (b) 10000011

   (c) 0xF7

   (d) 00000011

Examples: multiple choices

2. In the assembly programming, which of the following statements is true?
   (a) Use of macros can reduce code size.
   (b) Use of functions can reduce code execution time.
   (c) Not all functions can be converted into macros.
   (d) None of above.

Examples: Small Questions

1. Explain the concept of endianness. What endianness does AVR use? Please give an example.

Examples: Small Questions

2. Assume the contents in two registers, R16 and R17, are: R16=0x05 and R17=0xF9, respectively. What result will be generated by the AVR instruction *mulsu R17, R16?* Show the contents of its destination where the result is saved.

Examples: Small Questions

3. What is stack frame? Draw a memory map to show the basic structure of a stack frame. List AVR instructions that can access data in the stack frame.

Examples: Medium Questions

1. Consider the following AVR assembly code:

```
 .MACRO delay
loop:               subi @0, 1
                    sbci @1, 0
                    nop
                    nop
                    nop
                    nop
                    brne loop   ; the taken branch takes two cycles.
.ENDMACRO
```

All instructions in the program are 2 bytes long. 1) What is the size of the code in bytes? 2) What is the range of values of each parameter in this macro? 3) The code can generate a delay. What is the range of the delay? Assume the processor frequency is 8 Mhz.

Examples: Programming and Design

1. An array of ten 2-byte integers is stored in the AVR program memory. Write an assembly program to convert the array to the different endianness format and store it in the data memory.

   Your program must satisfy the following requirements:

   – Your program must use at least one function.
   – All local variables and parameters must be stored in the stack space.
   – You must describe the stack frame structure for the function used in your program

Thank You!

myExperience Survey

- Please participate
- If you have some feedback not covered by the Survey, please send your comments to me.
- Your feedback is much appreciated and will be considered for future improvement.