Practical 1 Review

ECE 362
https://engineering.purdue.edu/ee362/

© Rick
Lab Schedule this Week

• This week: Practical Exams in Lab
  - You may go to Open Lab / Office Hours at regular times
  - You should go to your normal lab section.
  - Other than that, please stay out.
    • No personal access to lab outside Office Hours
    • i.e., you may not enter lab from 2:30pm to 7pm or 10pm to 7:30am.
    • If you are an ECE 270 UTA, do not use the side door.
    • Why? Preparation and exceptions for students with accommodations.
Practical 1 Format

• Part 1: Concept questions
  – These will be similar in style to prelab/homework questions.
  – You get softcopy versions of the reference manuals.
  – This part is worth about 10% of the course grade.

• Part 2: Program the development board
  – We supply the board and do all wiring in advance.
  – We give you some kind of automatic grading interface.
  – This part is worth about 13% of the course grade.

• You will have a total of two hours to work.
  – Do either part first, go back and forth, etc.
  – When time is up, we will collect your exam and print your program.
You may bring...

- Nothing.
  - No calculator.
  - No phone.
  - No pencil.
  - No pen.

- You will place your belongings at the front of the room by the window before you go to your assigned lab station.
Instruction encoding/decoding

• Do you remember how to decode a 16-bit value to find the instruction?
• Do you remember how to take an instruction and encode it into a 16-bit value?
Machine Instructions

- Move instruction
- Arithmetic instructions
- Logical instructions
- Shift/rotate instructions
- Control flow instructions (B, B**, BL, BX)
- Load & Store instructions
  - Load literal
- Stack instructions (PUSH, POP, ADD, SUB)
- Why do some instructions have an ‘S’ suffix?
  - Because they _set_ the flags. (Except for the CMP instruction which has no suffix.)
Addressing Modes

• Register values
• Immediate values
• Some instructions support only one form.
  - e.g., ORRS R0,R1
• Some support both.
  - e.g., MOVS R0,R1; MOVS R1,#4
Assembler Directives

- .cpu
- .thumb
- .syntax
- .text
- .data
- .global symbol
- .word value
- .space size
- .string "…"
- .align boundary
- .equ name, replacement
Assembler Labels

- A label is a symbolic name for an address.
- An EQU is a symbolic name for anything.
- Neither one of these things causes space to be reserved in memory.
Translation from C to assembly

- Simple statements
- Representation of variables as registers
- Representation of variables in the data segment
  - How do you load and store a value to a word of memory in the data segment whose address is represented by a label?
- if-then-else
- do-while loops
- while loops
- for loops
Using the Stack

• How do subroutines work?
  – Conforming to the Application Binary Interface.
  – How do you pass parameters to a subroutine?
  – How do you return a value from a subroutine?
    • Even if a subroutine does not return a value, it must still return.

• How do push and pop work?
  – When and why do we need to use them?

• Growing and shrinking the stack
  – Where does it start?
  – Which way does it grow?
Recursive Subroutines

Would it be a problem to translate this function to assembly?

```c
int fn(int x, int y) {
  if (x >= y)
    return x-y;
  return x * fn(x+1,y-1) + y;
}
```

```assembly
.global fn
fn:
  push {r4,r5,lr}
  cmp r0,r1
  blt recurse
  subs r0,r1
  pop {r4,r5,pc}
recurrse:
  movs r4,r0
  movs r5,r1
  adds r0,#1
  subs r1,#1
  bl fn
  muls r0,r4
  adds r0,r5
  pop {r4,r5,pc}
```
Memory Reference Nuances

• Alignment
  – What does it mean? Where is it needed?

• Endianness
  – What does it mean? When does it matter?
General Purpose I/O

- Be familiar with the memory map and layout of control registers.
- What is the RCC?
  - What does it do?
  - How do you configure it?
- What are some of the GPIO control registers you will use and why?
  - How do you configure a pin to be an input?
  - How do you configure a pin to be an output?
  - How do you set a specific output pin high?
  - How do you set a specific output pin low?
  - How do you pull input pins high or low?
  - How do you set the output type? The output speed?
  - How do you choose an alternate function? Why? Which one?
Which pins do what?

- Which pin can be configured for DAC_OUT1?
  - Maybe you don’t remember, but you can look this up, right?
Interrupts

• What is an interrupt?
  – It’s a hardware-invoked subroutine.

• What is the one thing an ISR should always do?
  – Acknowledge the interrupt by checking and clearing the specific interrupt flag.
  – What happens if it doesn’t?

• What are the steps to enable an interrupt?

• What registers are saved before invoking an ISR?

• What happens if, while an ISR is running:
  – A higher priority interrupt occurs?
  – A lower priority interrupt occurs?