ECE 319K
Introduction to Embedded Systems

Lecture 6:
Exam 1 review

- Exam 1: **Thu-Feb 20, 2025 7:00PM - 8:30PM**

Agenda

- Exam 1
  - Exam 1, Thu-Feb 20, 2025 7:00PM - 8:30PM.
  - Closed book, no calculator.
  - See handout at end of each exam

- Topics
  - Memory access
  - Registers and stack
  - Arithmetic operations
  - Shift operations
  - Logic operations
  - Control structures
  - Functions
  - Arrays

Exam 1 What to study?

- Labs 1-3
  - Arrays in C and assembly
- Book chapters 1-3, eBook Chapters 1-3
  - See reading assignments for Labs 1-3
- Canvas Quizzes up through 10/4
- Old exams
  - Familiarize reference sheet with old exams
    - Instructions, pseudops, I/O registers
- AAPCS

Exam 1 What not to study?

- Divide algorithm
- Stepper motor, solid state relay
- .macro
- SysTick, TimerG,
- no pointers in C
  - Yes to pointers in assembly
- printf
- LED interface with ULN2003B driver
- No NZVC bits
  - Yes to CMP, ANDS..SUBS followed by Bxx
- .align
- Structs
- FSM

Memory Access

- **Signed or Unsigned**
- **8/16/32 bits**
  - LDRB
  - LDRSB
  - LDRH
  - LDRSH
  - LDR
  - STRB
  - STRH
  - STR

What do the following instructions do?

```
LDR   R1,[R2]
LDRB  R1,[R2,#2]
LDRSB R1,[R2,R3]
LDRH  R1,[R2,R3]
```

What is the difference?

```
LDR R0,=100
MOV R0,#100
```

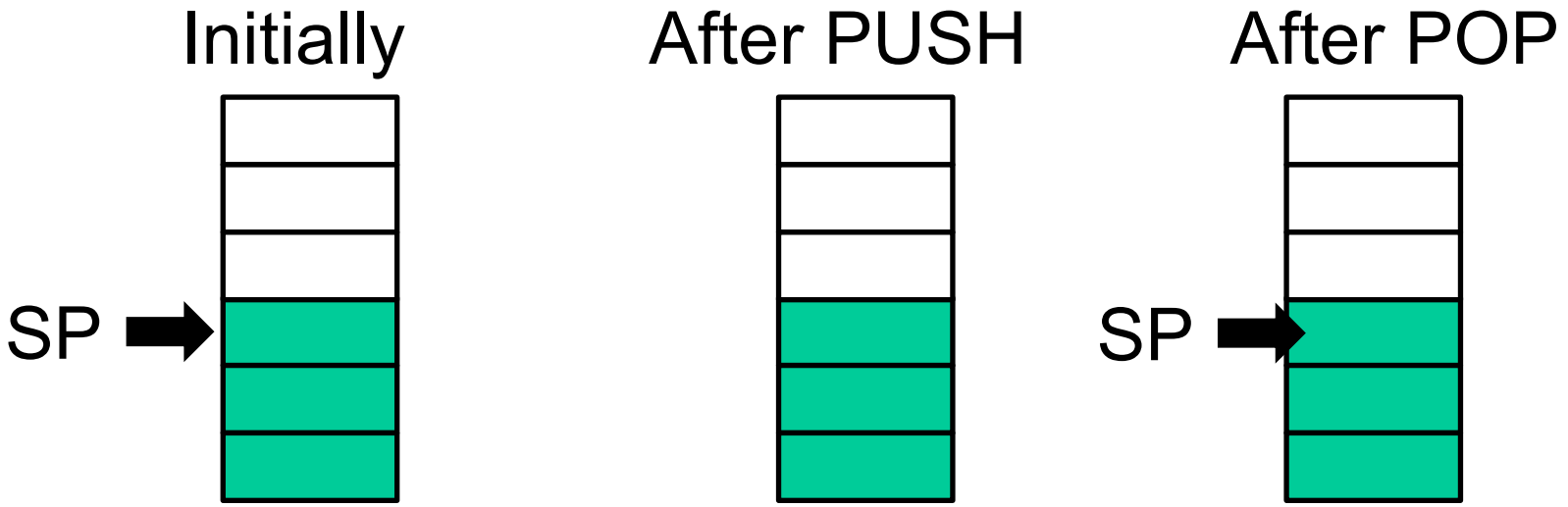
Stack access

- **SP points to data**
- **Push**
  - SP = SP-4
  - Store at SP
- **POP**
  - Read at SP
  - SP = SP+4
- **Order of data (one push or pop)**
  - Smaller register number at lower address
  - Larger register number at higher address

Activity

- **Initially, let R0=0, R1=1, R2=2**
- **Execute** `PUSH {R1,R2}`
- **Draw stack**

- **Execute** `POP {R0,R1}`
- **Draw stack**



Instapoll Lec 6

- **Initially, let R0=0, R1=1, R2=2, R3=3**
- **Execute**

```
PUSH {R1,R2}
PUSH {R3,R0}
POP {R0-R3}
```

- **What is in register ?**

AAPCS, Registers and stack

- **Save LR before calling another function**

```
Fun1: PUSH {R4,LR}
      BL   Fun2
      POP  {R4,PC}
```

- **Input parameters in R0,R1,R2,R3**
- **Output parameter in R0**
- **Balance the stack**
- `PUSH {R4,R5}` **same as** `PUSH {R5,R4}`
- **Freely use R0,R1,R2,R3,R12**
- **Save, use, restore R4-R11**

```
Fun3: PUSH {R4-R7}
      //stuff using R4-R7
      POP  {R4-R7}
      BX   LR
```

Arithmetic, logical, shift


```
ADDS        ANDS        LSLS
SUBS        ORRS        LSRS
RSBS        BICS        ASRS
MULS        EORS
```

How do you tell what operands are possible?

```
ADDS R0,R1,#3
ANDS R0,R1,#3 //not allowed (see handout)
MULS R0,R1,#3 //not allowed (see handout)
LSL R0,R1,#3
```

Control Operations

- **If-then**

```
CMP R0,#0
Bxx Target
```

```
    CMP R0,#0
    Bxx Yes
No: //not true
//
    B   done
Yes: //true
//
done:
```

- **If-then-else**

- **Do-while**

```
loop:
//body
    CMP R0,#0
    Bxx loop
```

- **While**

```
loop: CMP R0,#0
      Bxx done
//body
      B loop
done:
```

BEQ	BEQ
BNE	BNE
BLO	BLT
BLS	BLE
BHI	BGT
BHS	BGE

Special cases

- **If-then with bit mask**

```
MOVS R1,#4          MOVS R1,#4
ANDS R0,R1          ANDS R0,R1
BEQ  Target         BNE  Target
```

- **Do-while with counter**

```
loop:
//stuff
     SUBS R0,#1
     BNE  loop
```

Arrays

- **Base**
  - **Pointer passed in a register**
- **Precision**
  - **8 bits n=1 byte**
  - **16 bits n=2 bytes**
  - **32 bits n=4 bytes**
- **Size**
  - **Fixed and known**
  - **Passed in another register**
- **Value**
  - **Character**
  - **Unsigned integer**
  - **Signed integer**

Examples

- **Simple programs**
  - Clear, set, toggle, swap bits
  - Check if a bit is high or low
  - Logic, shift
  - Add, sub, multiply, and divide
- **Array programs**
  - Min, max, sum, product, difference, average
  - Search for element
  - Replace one element with another
  - Copy from one to another

Strategies

- **Understand the question**
  - What are the inputs?
  - What is the output?
- **Design**
  - Understand complexity
    - If-then
    - Single loop
    - Double loop
  - Pseudo code
- **Implementation**
  - AAPCS
- **Debug it**
  - Single step your code in your mind
  - Do the inputs generate proper outputs?