Lecture 14: Stack
Today’s Goals

• Learn how HCS12 stack functions
What is the Stack?

Last in, First out (LIFO)

• The stack is a last-in, first-out (LIFO) data structure.

• Two fundamental operations
  ▪ Push: add to the top of the list
  ▪ Pop/Pull: removes an item from the top of the list.

• Temporary variable storage
  ▪ When subroutines are called.

• Think books stacked on a table

• c.f. Queue
  ▪ First-in, first-out (FIFO) data structure
The Stack

Memory diagram of a stack

Available Stack Space

Top of Stack

Used Stack Space

Bottom of Stack
Important Concepts about Stacks

• A regular RAM section will be used. No hardware enforces this boundary.

• Much like variable-length array

• The stack grows toward lower addresses when a value is added.

• SP (Stack Pointer register) holds the address of the current index of the stack.
  ▪ SP moves up (decrements) when an item is added
  ▪ SP moves down (increments) when an item is removed
  ▪ Note: Addition/Removal happens after SP moves up/down
  ▪ All bytes in the stack space that are located at addresses lower than SP are considered unused
Important Concepts about Stacks

- **Depth:**
  - the number of bytes stored on the stack

- **Empty:**
  - If the depth is 0, we say the stack is empty.

- **Underflow**
  - Removing a byte from an empty stack

- **Overflow**
  - The depth is larger than the available stack space
Manipulating the Stack and SP

Adding/Removing an item to a stack

- **PSHx**
  - Push an item to the stack
  - X: A, B, C(CCR), D, X, Y
    1. S register is first decremented by the number of bytes to be pushed,
    2. then the value from the register is copied into the memory indicated by S.

- **PULx**
  - Pull or Pop an item from the stack
  - X: A, B, C(CCR), D, X, Y
    - One or two bytes (depends on the size of the register) at the top of the stack is/are copied into the specified register,
    - then S is incremented

- Note: PSHx and PULx (except for PULC) do not affect on CCR bits

- Again, there is no separate space for the stack. The stack is just a chunk of RAM. S register holds the current position.
Manipulating the Stack and SP

- **LDS**
  - Load the stack pointer
  - Typically use only immediate addressing mode.
  - Typically initialize the pointer at the beginning of a program.

- **LEAS**
  - Remember LEAX and LEAY?
  - Load Effective Address into S register
  - This is useful to manipulate stack pointer without pushing/popping values.
  - With negative offsets, create space on the top of the stack
  - With positive offsets, removes data from the top of the stack
Examples

Example 1

1: LDS #$3600
2: LDAA #$AA
3: LDAB #$BB
4: PSHA
5: PSHB
6: PULA
7: PULB
Examples

Example 2

1: LDS #$3600
2: LDAA #$AA
3: LDAB #$BB
4: PSHD
5: LDD #$CCDD
6: LEAS 2,SP
Questions?
Wrap-up

What we’ve learned

• Stack

• PUSx, PULx, LDS, LEAS
What to Come

• Subroutines