1

# Lecture 4: Addressing Modes

## Today's Goals

**Two major goals**

- Understand addressing modes so figure out how to use them.
  - If you don't get addressing modes, you will have a serious problem to complete this course.

- Learn how to use a program trace.

# Addressing Modes

## How to get effective addresses

- The operand of an instruction can use different methods for specifying data in the memory (=addressing modes).
  - If the data number is in registers (inside the microprocessor), a memory address is not needed.

- The HCS12 has six addressing modes
  - Extended (EXT)
  - Direct (DIR)
  - Inherent (INH)
  - Immediate (IMM)
  - Index (IDX)
  - Relative (REL) : Used only with branch instructions.

- Effective Address
  - The effective address is the location that holds the data to be used by the operation.
  - The operand is often used to construct the effective address.
  - An addressing mode tells the microprocessor the way of calculation to get the effective address.

# Extended Addressing (EXT)

## Also called Absolute Addressing

- Effective address:
  - No operation needed.
  - Extended addressing tells the full memory address.

- Format:
  - Two-byte hexadecimal number (4-digit) preceded with a $. Actually '$' simply means that the number is a hexadecimal number. (A number could be followed by 'h' excluding ''.

| | |
|---|---|
| | . . |
| 2000 | B6 |
| 2001 | 30 |
| 2002 | 00 |
| | . . |
| 3000 | 98 |
| | . . |

- Example:
  - (Assuming the instruction is stored at $2000)
  - LDAA    $3000
    - Load a byte value stored at address $3000 into the register A.
    - LDAA    *opr16a*        (M) ➔ A        EXT    B6 hh ll
    - 98 ➔ A

3

# Direct Addressing (DIR)

## Also called Zero-Paging Addressing

- Effective address:
  - This addressing mode only supplies the lower byte of the address.
  - Extend the one byte address to two-bytes by concatenating $00 to the beginning of the operand.
- Format:
  - One byte hexadecimal number (2-digit) preceded with a $.
- Example:
  - (Assuming the instruction is stored at $2000)
  - LDAA    $80
    - Load a byte value stored at address $0080 into the register A.
    - LDAA   *opr8a*      (M) ➔ A        DIR    96 dd
    - 98 ➔ A

| | |
|---|---|
| | . . |
| 0080 | 98 |
| | . . |
| 2000 | B6 |
| 2001 | 30 |
| | . . |

# Inherent Addressing (INH)

## Also called Implied Addressing

- Effective address:
  - No operation.

- Format:
  - No operand.

- Example:
  - (Assuming the instruction is stored at $2000)
  - INCA
    - Increase register A by 1
    - INCA              (A) + $01 ➔ A        INH    42

| | |
|---|---|
| | . . |
| 2000 | 42 |
| | . . |

# Immediate Addressing (IMM)

| | |
|---|---|
| | . |
| | . |
| 2000 | 86 |
| 2001 | 80 |
| | . |
| | . |

| | |
|---|---|
| | . |
| | . |
| 2000 | CC |
| 2001 | 03 |
| 2002 | E8 |
| | . |
| | . |

- Effective address:
  - No operation. The data itself is supplied as the operand.
- Format:
  - Number preceded with a #. '#' is followed by a number that is a <u>value</u> instead of an <u>address</u>!
- Example:
  - (Assuming the instruction is stored at $2000)
  - LDAA  #$80
    - Load a byte value(the operand itself) into the register A.
    - $80_{16} \rightarrow$ A
  - LDD  #1000
    - 1000 is $03E8_{16} \rightarrow$ D (meaning 03 → A and E8 → B)
- The size of an operand
  - Register A and B have one-byte immediate operands.
  - Register D, X, Y, SP, and PC have two-byte ones.

# Index Addressing (IDX, IDX1, IDX2)

- Effective Address
  - Add the operand as a signed number to the value in the X, Y, PC, or S registers.
- Format
  - Signed number, Register (X, Y, PC, or S)
- Example:
  - LDAA  0,X
    - The effective address is the value(=address) in register X. (=X+0)
  - LDD  -100,Y
    - The effective address is 100 lower than the value in Y. (=Y-100)
  - LDX  1000, Y
    - The effective address is 1000 higher than the value in Y. (=Y+1000)
- Notes:
  - The value in the specified register is not changed.
  - The smallest number of bits will be used to represent  the address.

# Index Addressing Postbytes

- An operand in the index addressing are called a **postbyte**.

- The postbyte tells the processor which two-byte register to be used as the base address, the size of the offset.

| Register | rr |
|----------|----|
| X | 00 |
| Y | 01 |
| SP | 10 |
| PC | 11 |

Postbyte for 5-bit Offset:    rr0nnnnn
Postbytes for 9-bit Offset:    111rr00n nnnnnnnn
Postbytes for 16-bit Offset:    111rr010 nnnnnnnn nnnnnnnn

---

# Index Addressing

## Examples

| Instruction | | Machine Code | | |
|-------------|------|--------------|----------|-----------|
| LDAA 4,Y | A6 | 44 | | |
| | | **01** 0 **00100** | | |
| LDD -100,X | EC | E1 | 9C | |
| | | 111 **00** 00 **1** | **10011100** | |
| LDX -1000,Y | EE | EA | FC | 18 |
| | | 111 **01** 010 | **1111 1100** | **0001 1000** |

| Register | rr |
|----------|----|
| X | 00 |
| Y | 01 |
| SP | 10 |
| PC | 11 |

Postbyte for 5-bit Offset:    rr0nnnnn
Postbytes for 9-bit Offset:    111rr00n nnnnnnnn
Postbytes for 16-bit Offset:    111rr010 nnnnnnnn nnnnnnnn

# Instruction Set

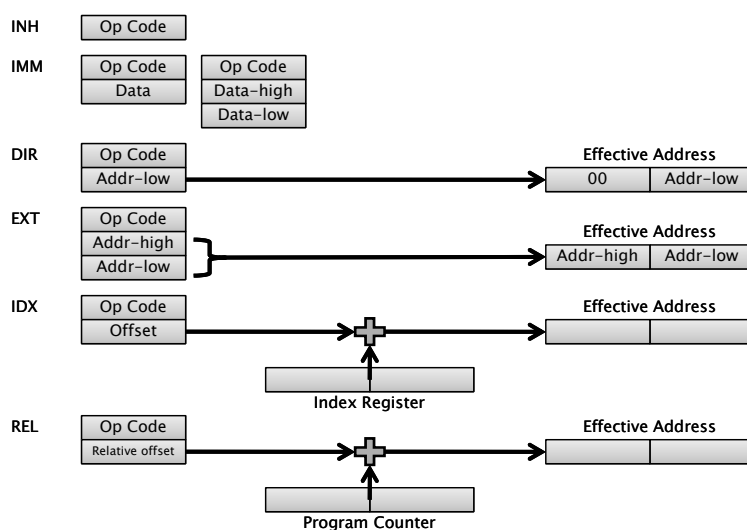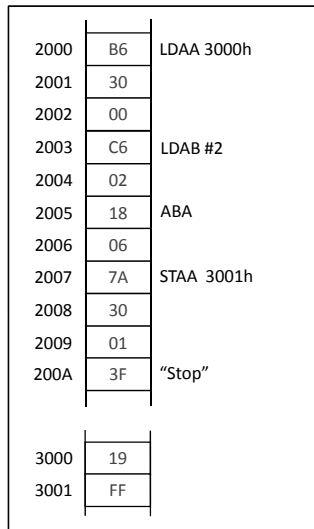| Source Form | Operation | Addr. Mode | Machine Coding | Access Detail | S X H I | N Z V C |
|---|---|---|---|---|---|---|
| LDAA #opr8i | (M) $\Rightarrow$ A | IMM | 86 ii | P | - - - - | $\Delta\,\Delta$ 1 0 |
| LDAA opr8a | Load Acc. A | DIR | 96 dd | rPf | | |
| LDAA opr16a | | EXT | B6 hh ii | rPO | | |
| LDAA oprx0_xysp | | IDX | A6 xb | rPf | | |
| LDAA oprx9,xysp | | IDX1 | A6 xb ff | rPO | | |
| LDAA oprx16,xysp | | IDX2 | A6 xb ee ff | frPP | | |

- Above is a portion of the entry for the LDAA instruction.

- Now, we can better understand information in the HCS12 instruction sets.

# Addressing Mode Summary
## How to Get an Effective Address

## Program Trace

| Address | Data | Instruction |
|---------|------|-------------|
| 2000 | B6 | LDAA 3000h |
| 2001 | 30 | |
| 2002 | 00 | |
| 2003 | C6 | LDAB #2 |
| 2004 | 02 | |
| 2005 | 18 | ABA |
| 2006 | 06 | |
| 2007 | 7A | STAA 3001h |
| 2008 | 30 | |
| 2009 | 01 | |
| 200A | 3F | "Stop" |
| | | |
| 3000 | 19 | |
| 3001 | FF | |

- A diagram showing the contents of the HCS12 memory which contains a program.

- A program trace shows the contents of the processor's registers as the program is executed.

- Very useful for debugging programs

## Program Trace

### Example

| Trace Line | Address | Instruction | PC | A | B |
|------------|---------|-------------|-----|-----|-----|
| 1 | 2000 | LDAA 3000h | 2003 | 19 | - |
| 2 | 2003 | LDAB #2 | 2005 | 19 | 02 |
| 3 | 2005 | ABA | 2007 | 1B | 02 |
| 4 | 2007 | STAA 3001h | 200A | 1B | 02 |
| 5 | 200A | "stop" | - | - | - |

8

# Program Trace

## Another Example

| Address | Value | Instruction |
|---|---|---|
| 2000 | CE | LDX #3001H |
| 2001 | 30 | |
| 2002 | 01 | |
| 2003 | EC | LDD 1,X |
| 2004 | 01 | |
| 2005 | 87 | CLRA |
| 2006 | 6C | STD -1,X |
| 2007 | 1F | |
| 2008 | 3F | "Stop" |
| | | |
| 3000 | EC | |
| 3001 | 27 | |
| 3002 | 45 | |
| 3003 | 99 | |

| Trace Line | Address | Instruction | PC | X | A | B |
|---|---|---|---|---|---|---|
| 1 | 2000 | LDX #3001h | 2003 | 3001 | - | - |
| 2 | 2003 | LDD 1,X | 2005 | 3001 | 45 | 99 |
| 3 | 2005 | INCB | 2006 | 3001 | 45 | 9A |
| 4 | 2007 | STD -1,X | 2009 | 3001 | 00 | 9A |
| 5 | 2009 | "STOP" | - | - | - | - |

- X requires a 2-byte operand with immediate addressing since it is a 2-byte register.

- Note that using indexed addressing to load/store register D does not change the value in register X.

- What are the values in memory locations from 3000h to 3003h after the program is done executing? **45-9A-45-99**

# Questions?

## Wrap-up

**What we've learned**

- Five addressing modes

- Program trace

## What to Come

- Unconditional branches

- Relative addressing mode