Introduction to Microcontrollers III

Timing Functions

Delay5u.a11, Delay1m.a11

μp Laboratory #3

µP Laboratory #2 Hints

Data Entry :

- Use the pushbutton routine from count.a11 or count_br.a11 (WAIT0 and WAIT1 loops)
- Consider using Indexed addressing for entering data loops
- Store numbers to $D000-$D007
- Display numbers to output PortB ($1004)

μP Laboratory #2 Hints

Data Sum :

- Assume number located at $D000-$D007
- Consider using Indexed addressing for accessing each memory location
- Use AccA or AccB for temporary storage of sum

μP Laboratory #2 Hints

Data Sum :

- Assume number located at $D000-$D007
- Need two loops for sorting
- Consider using Indexed addressing for your loops

Sort Routine

Initialize counters

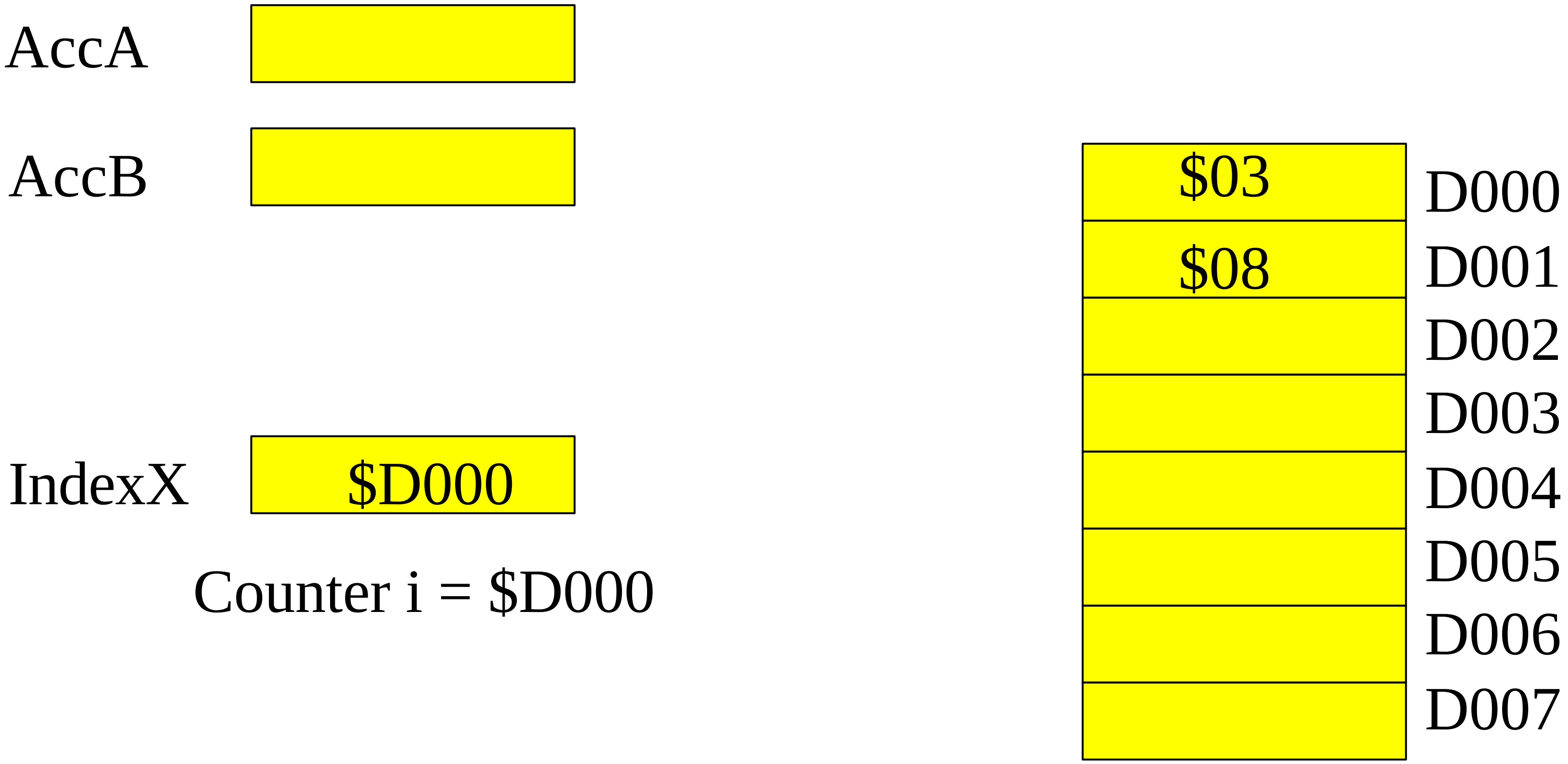
Compare contents of location i with i+1

If contents of i is less than contents of i+1, swap; otherwise increment counter

If inner loop is done increment outer loop counter

If outer loop is done, end; otherwise reset inner loop counter and begin again

Swap Routine



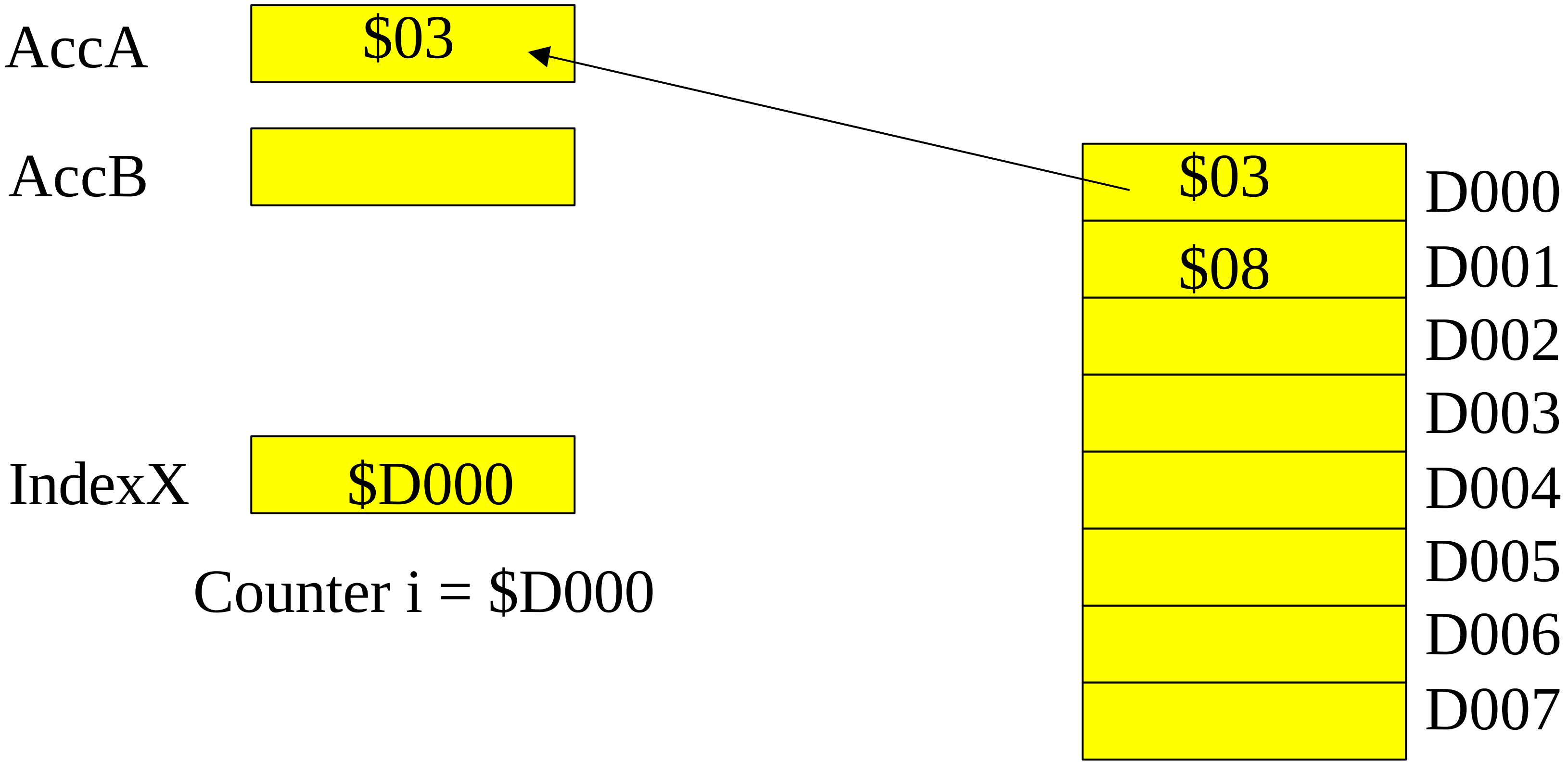
AccA	
AccB	
IndexX	$D000

Counter i = $D000

$03	D000
$08	D001
	D002
	D003
	D004
	D005
	D006
	D007

Swap Routine

ldaa 0,x



AccA $03

AccB

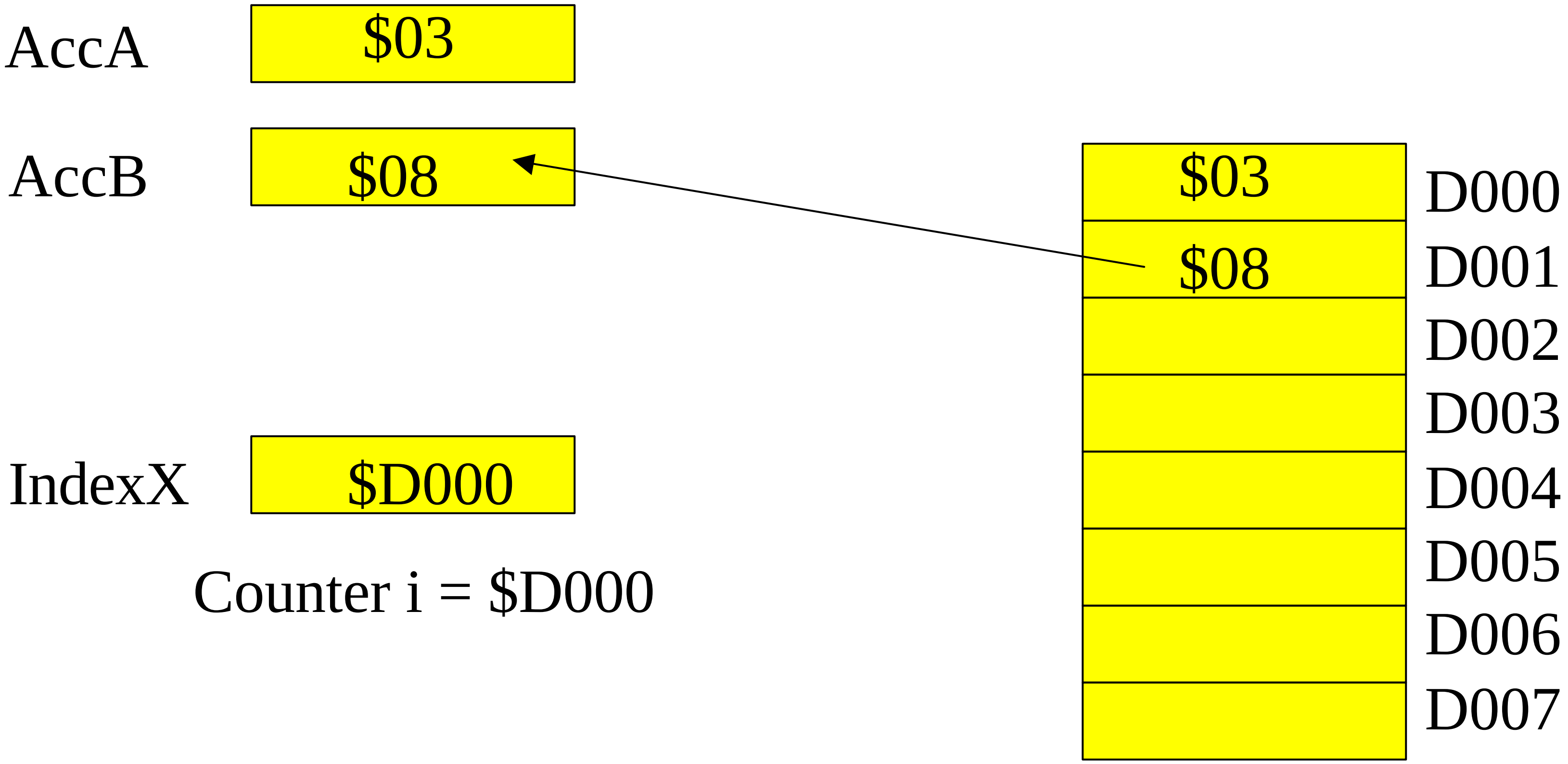
IndexX $D000

Counter i = $D000

$03	D000
$08	D001
	D002
	D003
	D004
	D005
	D006
	D007

Swap Routine

ldaa 1,x



AccA $03

AccB $08

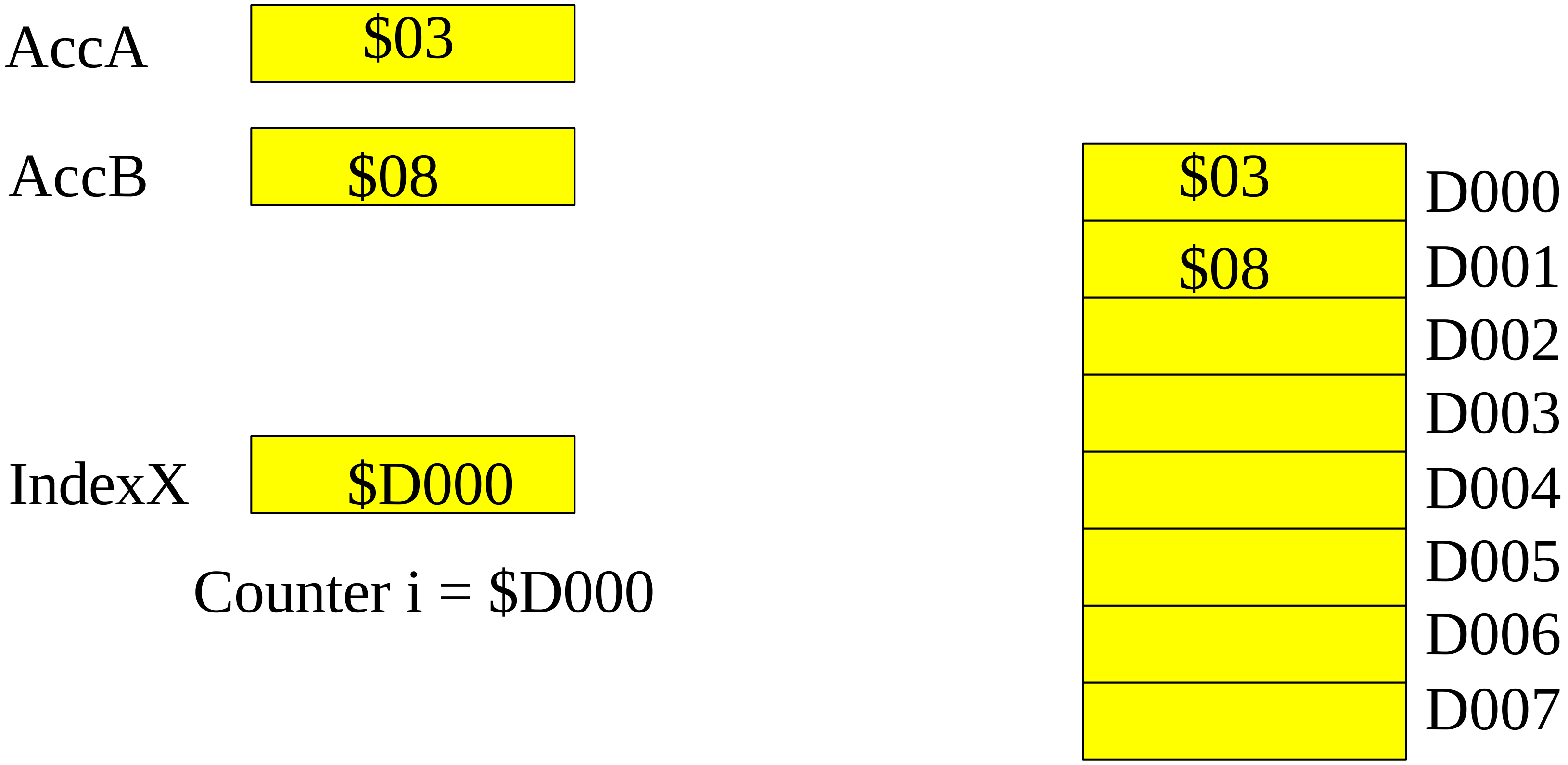
IndexX $D000

Counter i = $D000

$03	D000
$08	D001
	D002
	D003
	D004
	D005
	D006
	D007

Swap Routine

Compare AccB to AccA



AccA $03

AccB $08

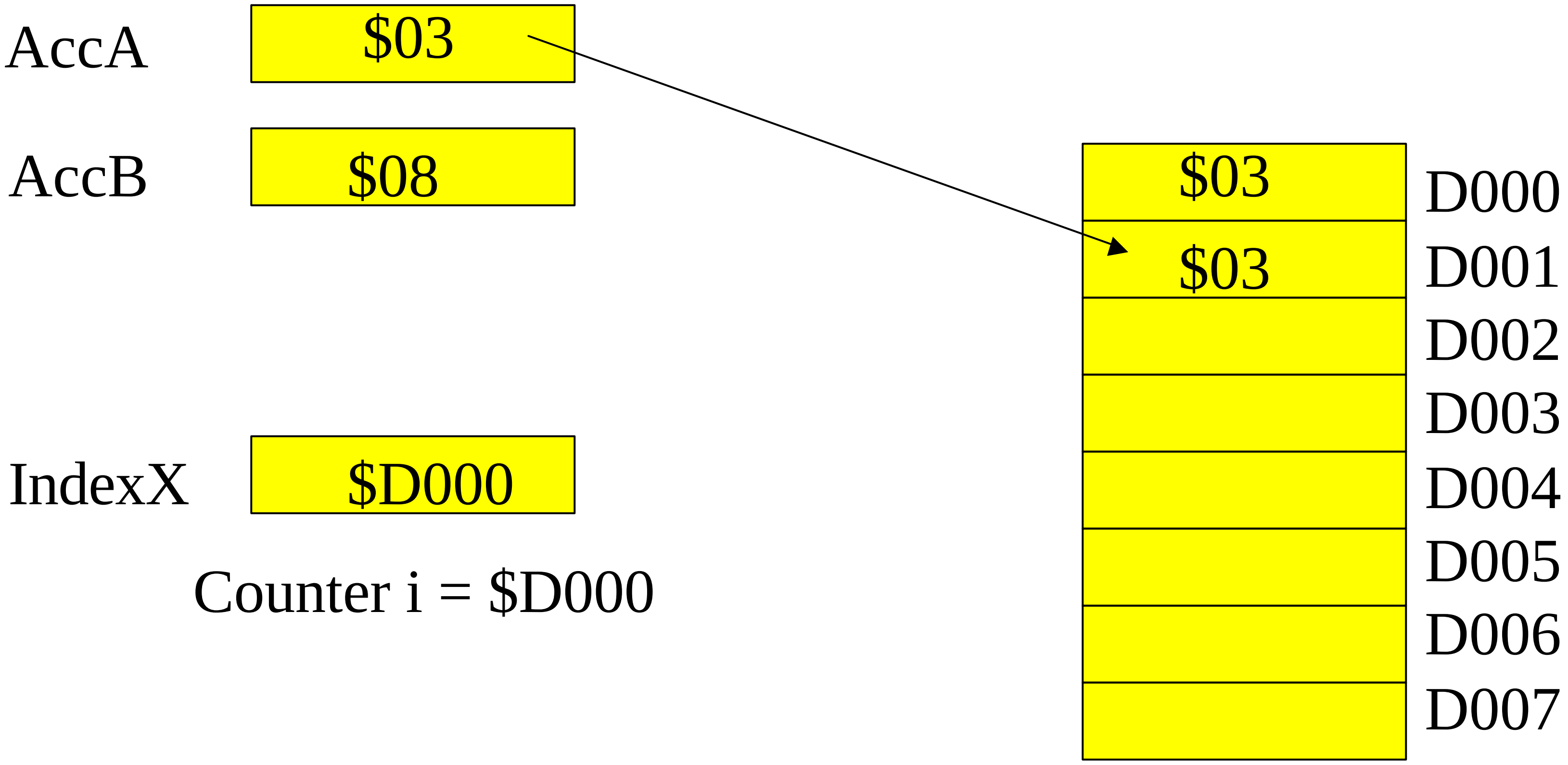
IndexX $D000

Counter i = $D000

$03	D000
$08	D001
	D002
	D003
	D004
	D005
	D006
	D007

Swap Routine

staa 1,x



AccA $03

AccB $08

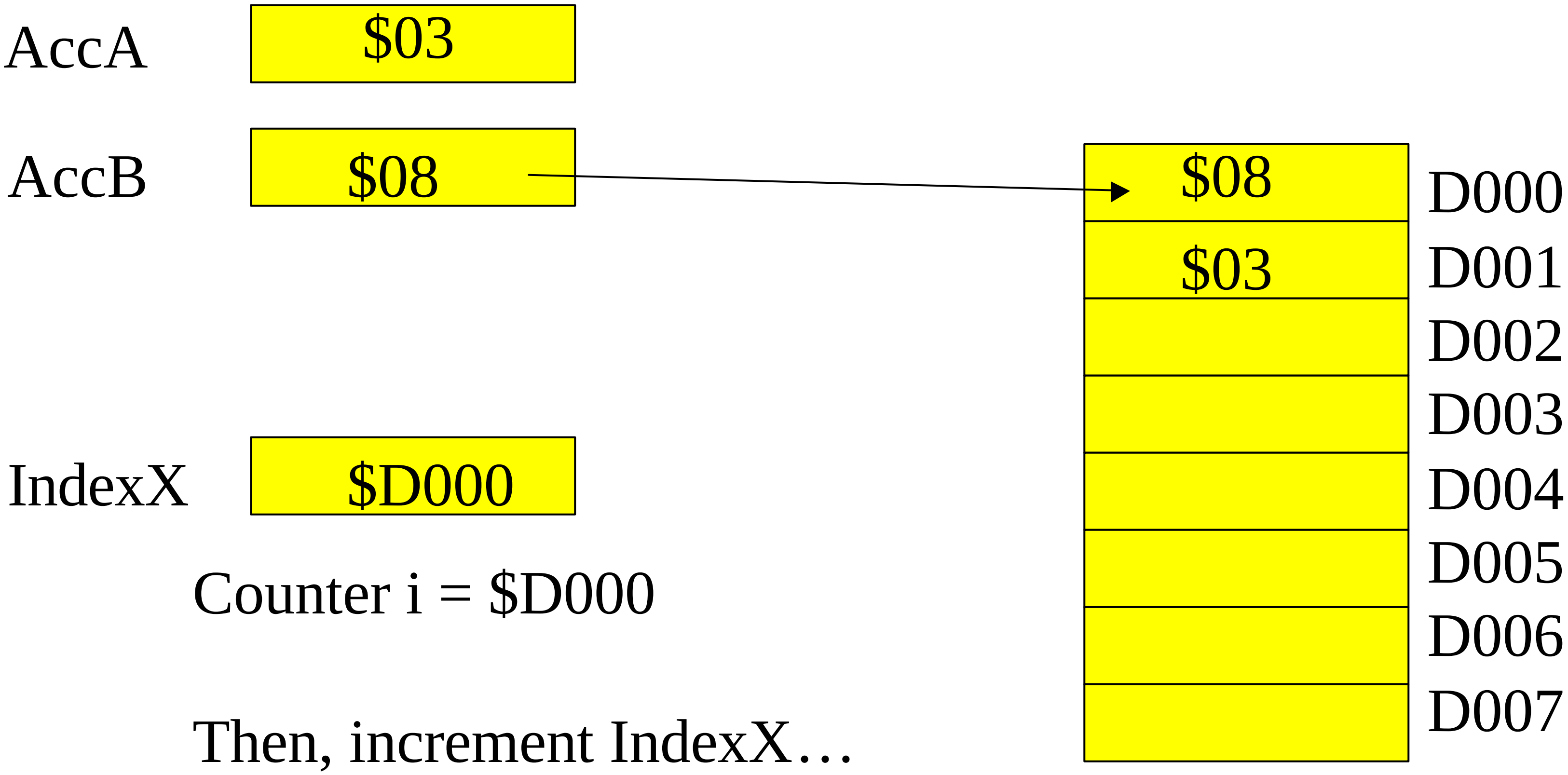
IndexX $D000

Counter i = $D000

$03	D000
$03	D001
	D002
	D003
	D004
	D005
	D006
	D007

Swap Routine

stab 0,x



AccA $03

AccB $08

IndexX $D000

$08	D000
$03	D001
	D002
	D003
	D004
	D005
	D006
	D007

Counter i = $D000

Then, increment IndexX…

What about timing functions?

- Sometimes you want to put a delay in your program.The HC11 has advanced features that use a real-time clock. Refer to chapter 10 of the HC11 Reference Manual for information on usage.
- An alternative is the use of delay subroutines: delay5u.a11 and delay1m.a11

Timer Application 1: delay5u.a11

Delay = X * 5 µs

Load X with the number of times you want to delay for 5 µs.

```
*****************************************************
* DELAY5U.A11
* AUTHORS                        DATE      COMMENTS
* JACOBSON/SEVCIK                2/26/90   VERSION 1.0
*
* DESCRIPTION
*  THIS ROUTINE GENERATES INTERNAL DELAYS IN
*  MULTIPLES OF FIVE (5) MICRO-SECONDS. THE
*  USER ENTERS A MULTIPLIER (16-BIT) INTO THE
*  X INDEX REGISTER WHICH DETERMINES THE
     NUMBER
*  OF FIVE MICRO-SECOND INTERVALS
*
* PARAMETERS
*  - X REGISTER CONTAINS MULTIPLIER
*  - INTERRUPTS ARE NOT AFFECTED
*  - SHORTEST DELAY IS 10 us (X < 3)
*  - RESOLUTION IS 5 us
*  - MAXIMUM DELAY IS 327680 us (X = 64K)
****************************************************
```

```
          ORG $COOO
          ldx #1000           ;FOR TESTING
          JSR DELAY5U         ;CALL ROUTINE TO TEST
          JMP $E00A           ;JUMP TO BUFFALO WHEN DONE
DELAY5U:  DEX                 ;CORRECT FOR JSR/RTS
          DEX                 ;OVERHEAD
          NOP
          NOP
DELWT1:   DEX                 ;DECREMENT MULTIPLIER
          NOP
          NOP
          BNE DELWT1
          RTS
***********************************************
* END DELAY5U
***********************************************
```

This program uses the JSR instruction to call the subroutine: DELAY5U.

The actual subroutine consists of loops of instructions that just take up computer time.

Timer Application 2: delay1m.a11

Delay = X * 1 ms

Load X with the number of times you want to delay for 1 ms.

```
************************************************
* DELAY1M.A11      N*1ms Delay Routine
* AUTHORS                          DATE       COMMENTS
* JACOBSON/SEVCIK                  2/26/90    VERSION 1.0
*
* DESCRIPTION
*  THIS ROUTINE GENERATES INTERNAL DELAYS IN
*  MULTIPLES OF ONE (1) MILLI-SECOND. THE
*  USER ENTERS THE DURATION OF THE DELAY
*  (ms) INTO THE X REGISTER
*
* PARAMETERS
*  - X REGISTER CONTAINS DURATION (ms)
*  - INTERRUPTS ARE NOT AFFECTED
*  - SHORTEST DELAY IS 1 ms (X =1)
*  - RESOLUTION IS 1 ms
*  - MAXIMUM DELAY IS 655,36 ms (X = 0)
*************************************************
   **
*
```

```
TEST ROUTINE
*   user must enter a value into X, then run
        ORG $COOO
* change operand of next instruction to change
* the delay
TEST            ldx #100        ;FOR 100 ms DELAY
        JSR DELAY1M     ;CALL ROUTINE TO TEST
        JMP $E00A       ;JUMP TO BUFFALO WHEN DONE
**************************************************
* DELAY1M N*1ms subroutine
**************************************************
DELAY1M:PSHA
* Primary Loop
DELWT2  LDAA #199;199 * 2ND LOOP = 1ms
        NOP
DELWT3  DECA                    ;SECONDARY LOOP = 1ms/199
        NOP
        BRN DELWT3      ;BRANCH NEVER = 3 CYCLE NOP
        BNE DELWT3      ;CONTINUE UNTIL 199 --> 0

        DEX             ;# OF 1ms LOOPS
        BNE DELWT2      ;CONTINUE UNTIL IX = 0

        PULA
        RTS
***********************************************
* END DELAY1M
***********************************************
```

This program uses the JSR instruction to call the subroutine: DELAY1M.

The actual subroutine consists of 2 loops of instructions that just take up computer time. Notice: PSHA, PULA

Laboratory μP3: Count-down Timer

Pre-lab:

Design a circuit using the HC11EVB that will meet the following specs:

1. A two-digit BCD number will be entered.
2. Display the number on 7 segment displays as it counts down to zero @ 1.00Hz
3. Make an audible noise for the last 1 second before reaching zero.
4. When the number reaches zero, drive a relay closed (Output an active high signal)
5. Use the HC11EVb as the primary controller.
6. Use a minimum number of extra Ics
7. The two-digit bumber will be entered using an 8-position dip switch of two BCD switches.

Pre-compile all source code. Bring source code listings (on paper) and floppy disk containing the files to lab. The files should be error free at the beginning of the lab period.

Include flow charts for your source code.