

IBCM

“Itty Bitty Computing Machine” Part 2

Fetch Execute Cycle

```
while (power is on) {  
    IR := mem[PC]  
    PC := PC + 1  
    execute instruction in IR  
}
```

Note: PC = program counter
 IR = instruction register

Memory

Address

00	3000
01	5000
02	6006
03	8003
04	A000
05	4000
06	F000

PC

IR

Accum

Memory

Address

00

01

02

03

04

05

06



PC



IR



Accum



Writing IBCM Code

1. Write High Level Pseudo Code
 - **For (i=1;i<max; i++)**
 - ...
2. Translate into IBCM Symbolic Instructions
 - **LOAD ONE**
 - STORE I**
3. Test code by hand
 - **(step thru code)**
4. Encode into machine code
 - **3016**
 - 4005**
5. Load Machine code into Simulator and run.
 - **Run!**

How Would You Code This?

```
if B
    S1;
else
    S2;
```

How Would You Code This?

```
while(B)  
  S;
```

IBCM Code to Sum the digits from 1 to N

Accessing Arrays in IBCM



$$S = S + A[i]$$

<u>mem</u>	<u>locn</u>	<u>label</u>	<u>op</u>	<u>addr</u>	<u>comments</u>
C00A	00		jmp	start	skip around the variables
0000	01	i	dw	0	int i
0000	02	s	dw	0	int s
0000	03	N	dw	0	int N
0001	04	one	dw	1	
0000	05	zero	dw	0	
0000	06				leave space for changes
0000	07				
0000	08				
0000	09				
1000	0A	start	readH		read N
4003	0B		store	N	
3004	0C		load	one	i = 1
4001	0D		store	i	
3005	0E		load	zero	s = 0
4002	0F		store	s	
3003	10	loop	load	N	if (i > N) goto xit
6001	11		sub	i	
E01A	12		jmpL	xit	
3002	13		load	s	s += i
5001	14		add	i	
4002	15		store	s	
3001	16		load	i	i += 1
5004	17		add	one	
4001	18		store	i	
C010	19		jmp	loop	goto loop
3002	1A	xit	load	s	print s
1800	1B		printH		
0000	1C		halt		halt

<u>mem</u>	<u>locn</u>	<u>label</u>	<u>op</u>	<u>addr</u>	<u>comments</u>
C00A	00		jmp	start	skip around the variables
0000	01	i	dw	0	int i
0000	02	s	dw	0	int s
0000	03	a	dw	0	int a[]
0000	04	n	dw	0	
0000	05	zero	dw	0	
0001	06	one	dw	1	
5000	07	adit	dw	5000	
0000	08				leave space for changes
0000	09				
1000	0A	start	readH		read array address
4003	0B		store	a	
1000	0C		readH		read array size
4004	0D		store	n	
3005	0E		load	zero	i = 0; s = 0;
4001	0F		store	i	
4002	10		store	s	
3004	11	loop	load	n	if (i >= N) goto xit
6001	12		sub	i	
E020	13		jmpL	xit	
D020	14		jmpe	xit	
3007	15		load	adit	form the instruction to add a[i]
5003	16		add	a	
5001	17		add	i	
401A	18		store	doit	plant the instruction into the program
3002	19		load	s	s += a[i]
0000	1A	doit	dw	0	
4002	1B		store	s	
3001	1C		load	i	i += 1
5006	1D		add	one	
4001	1E		store	i	
C011	1F		jmp	loop	goto loop
10/19/2 3002	20	xit	load	s	print s
1800	21		printH		
0000	22		halt		