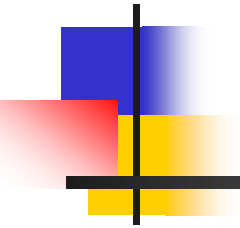
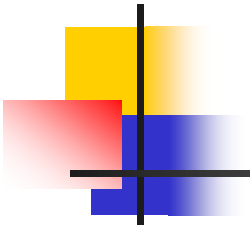


CPU Design Workshop

Silicon Valley
Forth Interest Group

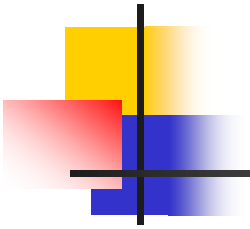
April 23, 2016

Chen-Hanson Ting



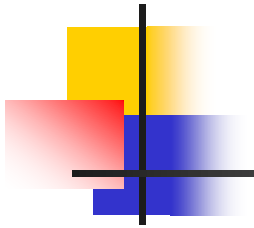
Summary

- Diamond IDE Demo
- 8086eForth 2.03
- Review of CPU Architectures
  - Historical: von Neuman, 1401, 701, Cardiac
  - Mini: PDP1, PDP8, PDP11, Nova
  - RISC: Sparc, MIPS, ARM



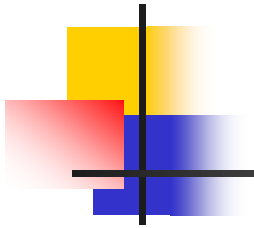
Synthesis of VHDL Files

- ep80_chip.vhd
  - ep80.vhd
  - uart80.vhd
  - gpio80.vhd
  - ram_memory.vhd



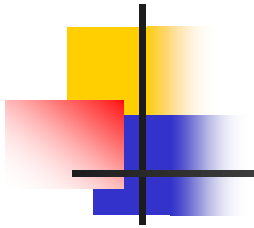
Programming FPGA

- Download FPGA image
- HyperTerminal Interface
- eForth experiments



Assembly eForth

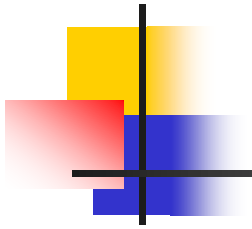
- Assemble 80ef203.asm
- Convert 80ef203.exe to ep203.mem
- Generate ram_memory.vhd
- Synthesis and programmer



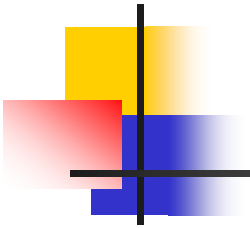
Reveal Debugger

- Reveal Inserter
- Reveal Analyzer

8086eForth 1.01

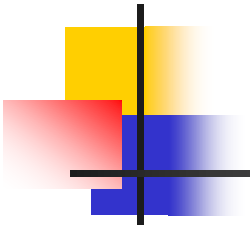


- Code word: machine instructions
- Colon word: CALL doList, token list
- User variable: CALL doUser, offset
  - : doUser R> @ UP @ + ;
- Variable: CALL doVar, value
  - : doVar R> ;



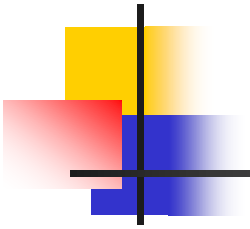
8086eForth 2.03

- Code word: machine instructions
- Colon word: CALL doList, token list
- User variable: CALL @, address
- Variable: CALL nextStep, value
- Constant: CALL @, value



CPU Architectures

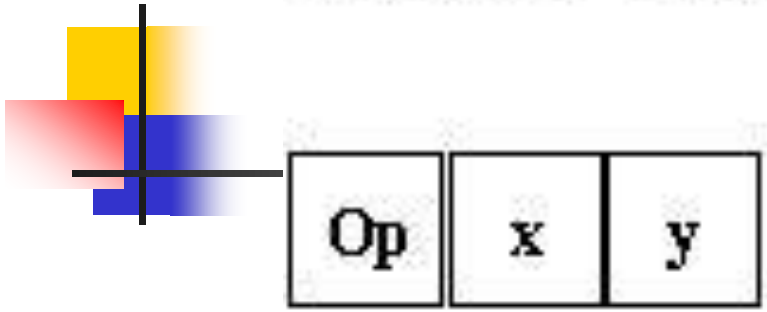
- Cardiac
- Von Neumann
- IBM 701, IBM 360
- PDP1, PDP8, PDP11
- Nova
- Sparc, MIPS, ARM



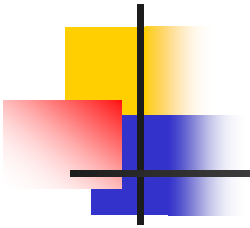
Cardiac Computer

- 3-digit decimal numbers
- Memory: 100 cells of 3 decimal digits
- Accumulator: 3 digits with – sign
- 1 input port and 1 output port
- 10 instructions

Cardiac Computer



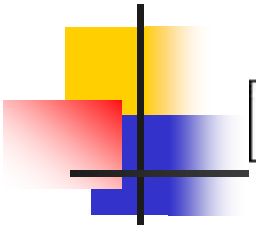
0 INP input
1 CLA clear and add
2 ADD add
3 TAC test and jump if negative
4 SFT shift x places left and y places right
5 OUT output
6 STO store
7 SUB subtract
8 JMP jump
9 HRS halt and reset



Von Neumann

- 40-bit binary words
- 2 registers AC and MQ
- 20-bit instructions
  - 12-bit address
  - 8-bit Opcode
- 21 instructions

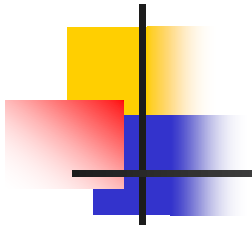
Von Naumann Machine



Left Address	Left Opcode	Rigth Address	Right Opcode

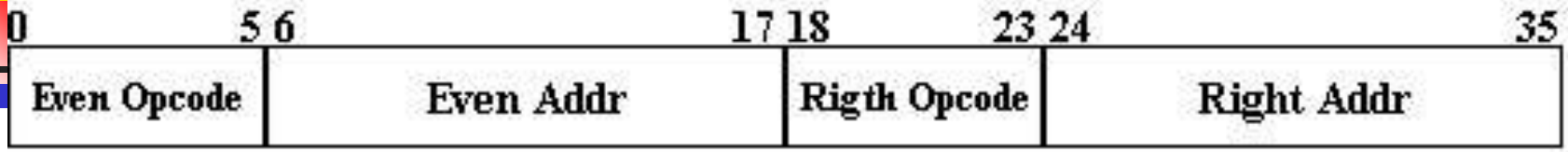
1 Move MQ to AC
2 Store AC to M
3 Store Left Addr to M Left
4 Store Left Addr to M Right
5 Load M to AC
6 Load -M to AC
7 Load |M| to AC
8 Load -|M| to AC
9 Add M to AC
10 Subtract M from AC
11 Add |M| to AC
12 Subtract |M| from AC
13 Move M to MQ
14 Multiply, M*MQ to AC,MQ
15 Divide, AC/M, Quot to MQ, Rem to AC
16 AC*2
17 AC/2
18 Jump to M Left
19 Jump to M Right
20 If AC>=0, jump to M Right
21 If AC>=0, jump to M Left

IBM 701

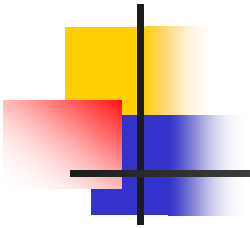


- 36-bit binary words
- 2 registers AC and MQ
- 18-bit instructions
  - 6-bit opcode
  - 12-Bit address
- 33 instructions

IBM 701 Instructions



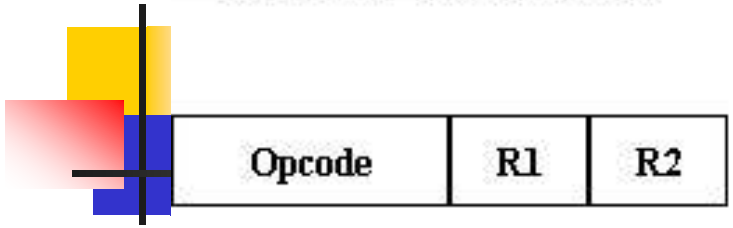
0 STOP	11 ADD AB	21 L RIGHT
1 TR	12 STORE	22 A LEFT
2 TR OV	-13 RXTR	23 A RIGHT
3 TR +	13 STORE A	24 READ
4 TR 0	14 STORE MQ	25 READ B
5 SUB	15 LOAD MQ	26 WRITE
6 R SUB	16 MPY	27 WRITE EOF
7 SUB AB	17 MPY R	28 REWIND
8 NOOP	18 DIV	29 SET DR
9 ADD	19 RND	30 SENSE
10 R ADD	20 L LEFT	31 COPY



IBM 360

- 32-bit binary words
- 16 32-bit registers and 4 64-bit floating point registers
- 8-bit instruction opcodes
- 2-6 byte variable length instructions
  - Integer instructions
  - BCD instructions
  - Floating point instructions

IBM 360 Instructions



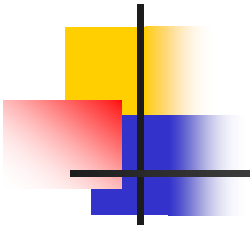
Opcode	R1	R2

Opcode	R1	X1	B2	D2

Opcode	R1	R3	B2	D2

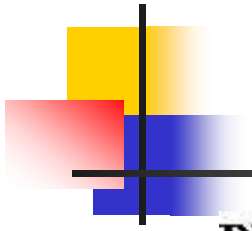
Opcode	I2	B1	B1	D1

Opcode	L1	L2	B1	D1	B2	D2



PDP-1

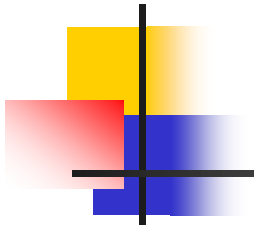
- 4096 18-bit core memory
- 2 registers AC and MQ
- 18-bit instructions
  - 5-bit opcode
  - 1-bit indirection
  - 12-bit address



PDP-1 Instruction Set

Op Code	I	Address

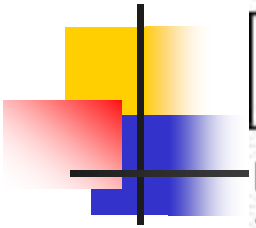
00	10 xct	20 lac	30 dip	40 add	50 sad	60 jmp	70 law
02 and	12	22	32 dio	42 sub	52 sas	62 jsp	72 ue
04 ior	14	24 dac	34 dzm	44 idx	54 mul	64 skp	74
06 xor	16 cal	26 dap	36 *	46 isp	56 div	66 sft	76 opr



PDP-8

- 4096 12-bit core memory
- 2 12-bit registers AC and MQ
- 6 Regular instructions
- 3 Microcode instructions

PDP-8 Instruction Set



Op Code	I	M	Offset

0 AND 3 DCA
1 TAD 4 JMS
2 ISZ 5 JMP

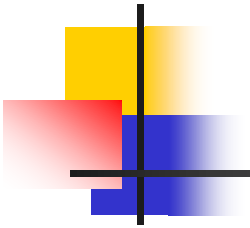
110	Device Number	Op Code

0 Interrupt System
3 Keyboard
4 Printer

1110	CLA	CLL	CMA	CML	RAR	RAL	0/1	IAC

1111	CLA	SMA	SZA	SZL	0/1	OSR	HLT	0

1111	CLA	MQA		MQL				1



PDP-11

- 16-bit core memory
- 8 16-bit registers
- Very powerful addressing modes
  - Direct, indirect and immediate
  - Pre-decrement and post-increment
  - Memory mapped IO
  - PC relative

PDP 11 Instruction Set

B	OP Code	Dest-Mod	Dest-Reg

B050 CLR	B060 ROT		0 Ri		0
B051 COM	B061 ROL		1 (Ri)+		1
B052 INC	B062 ASR		2 -(Ri)		2 #n
B053DEC	B063 ASL	Mod for	3 n(Ri)	Mod for	3 @#n
B054 NEG	B064	Registers	4 Ri@	PC (R7)	4
B055 ADC	B065		5 (Ri)+@		5
B056 SBC	B066		6 -(Ri)@		6 n
B057 TST	0067 SXT		7 n(Ri)@		7 @n

B	OP Code	Src	Dest

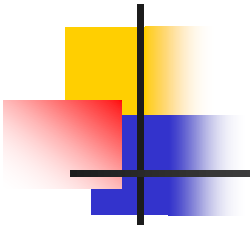
B1 MOV
B2 CMP
B3 BIT
B4 BIC
B5 BIS

0	OP Code	Src	Dest

070 MUL	073 ASHC
071 DIV	074 XOR
072 ASH	077 SOB

OP Code	Displacement

0000	0020 BGE	1000 BPL	1020 BVC	0001DD JMP
0004 BR	0024 BLT	1004 BMI	1024 BVS	004RDD JSR
0010 BNE	0030 BGT	1010 BHI	1030 BCC	00020R RTS
0014 BEQ	0034 BLR	1014 BLOS	1034 BCS	



Data General Nova

- 16-bit core memory
- 4 16-bit registers
- Very tight CPU design
  - Direct, indirect and immediate
  - Pre-decrement and post-increment
  - Memory mapped IO
  - PC relative

NOVA Instruction Set

0	00	Function	I	Index	Displacement

00 JMP
01 JSR
10 ISZ
11 DSZ

00 Page Zero
01 Disp+PC
10 Disp+AC2
10 Disp+AC3

0	Function	AC	I	Index	

01 LDA
10 STA

00 Page Zero
01 Disp+PC
10 Disp+AC2
10 Disp+AC3

0	11	AC	Transfer	Control	IO Device

000 NIO
001 DIA
010 DOA
011 DIB
100 DOB
101 DIC
110 DOC
111 SKP

00 NOP
01 SBCD
10 CBCD
11 Pulse

00 SKPBN
01 SKPBZ
10 SKPDN
10 SKPDZ

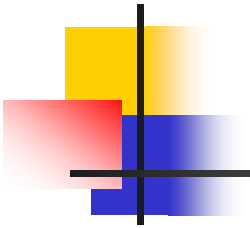
1	SAC	DAC	Function	Shift	Constrol	No Load	Skip

000 COM
001 NEG
010 MOV
011 INC
100 ADC
101 SUB
110 ADD
111 AND

00 NOP
01 ROL
10 ROR
11 SWAP

00 NOP
01 ZC
10 OC
11 CC

000 NOP
001 SZC
010 SNC
100 SZR
101 SNR
110 SEZ
111 SBN



SPARC

- 32-bit instructions and words
- 32 32-bit registers
- Load / Store
- 5 types of instructions

SPARC Instruction Set

00	a	COND	010	DISP22

0 BN	4 BLEU	8 BA	C BGU
1 BE	5 BLU	9 BNE	D BCC
2 BLE	6 BNEG	A BG	E BPOS
3 BL	7 BVS	B BGE	F BVC

00	RD	100	SETHI CONST22

01	CALL ADDR30

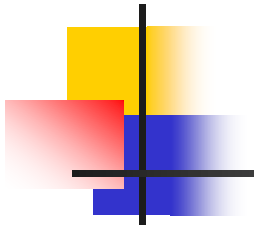
10	RD	OP2	RS1	0		RS2
				1	CONST13	

00 ADD	08	10 ADDCC	18
01 AND	09	11 ANDCC	19
02 OR	0A UMUL	12 ORCC	1A UMULCC
03 XOR	0B SMUL	13XORCC	1B SMULCC
04 SUB	0C	14SUBCC	1C
05 ANDN	0D	15 ANDNCC	1D
06 ORN	0E UDIV	16 ORNCC	1E UDIVCC
07 XNOR	0F SDIV	17 XNORCC	1F SDIVCC

25 SLL
26 SRL
27 SRA

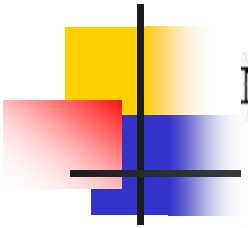
11	RD	OP2	RS1	0		RS2
				1	CONST13	

00 LD	04 ST	08
01 LDUB	05 STB	09LDSB
02 LDUH	06 STH	0A LDSH
03 LDD	07 STD	

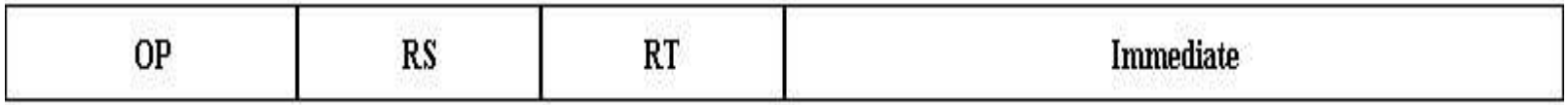


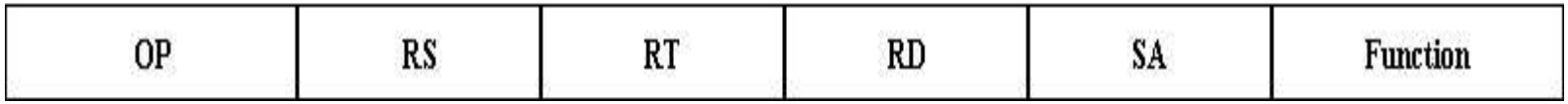
MIPS

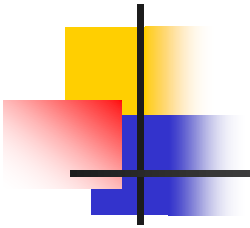
- 32-bit instructions and words
- 32 32-bit registers
- Load / Store
- 5 types of instructions



MIPS R2000 Instructions







MIPS Instructions

Arithmetic: add, addi, addiu, addu, sub, subu

Logic: and, andi, nor, or, ori, xor, xori

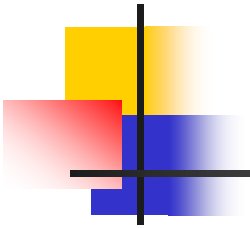
Shift: sll, sra, srl, sllv, srav, srlv

Comparison: slt, sltu, slti, sltiu

Load/Store: lui, lb, lbu, lh, lhu, lw, sb, sh, sw

Branch: beq, bne, bgez, bgezal, bgtz, blez, bltzal, bltz

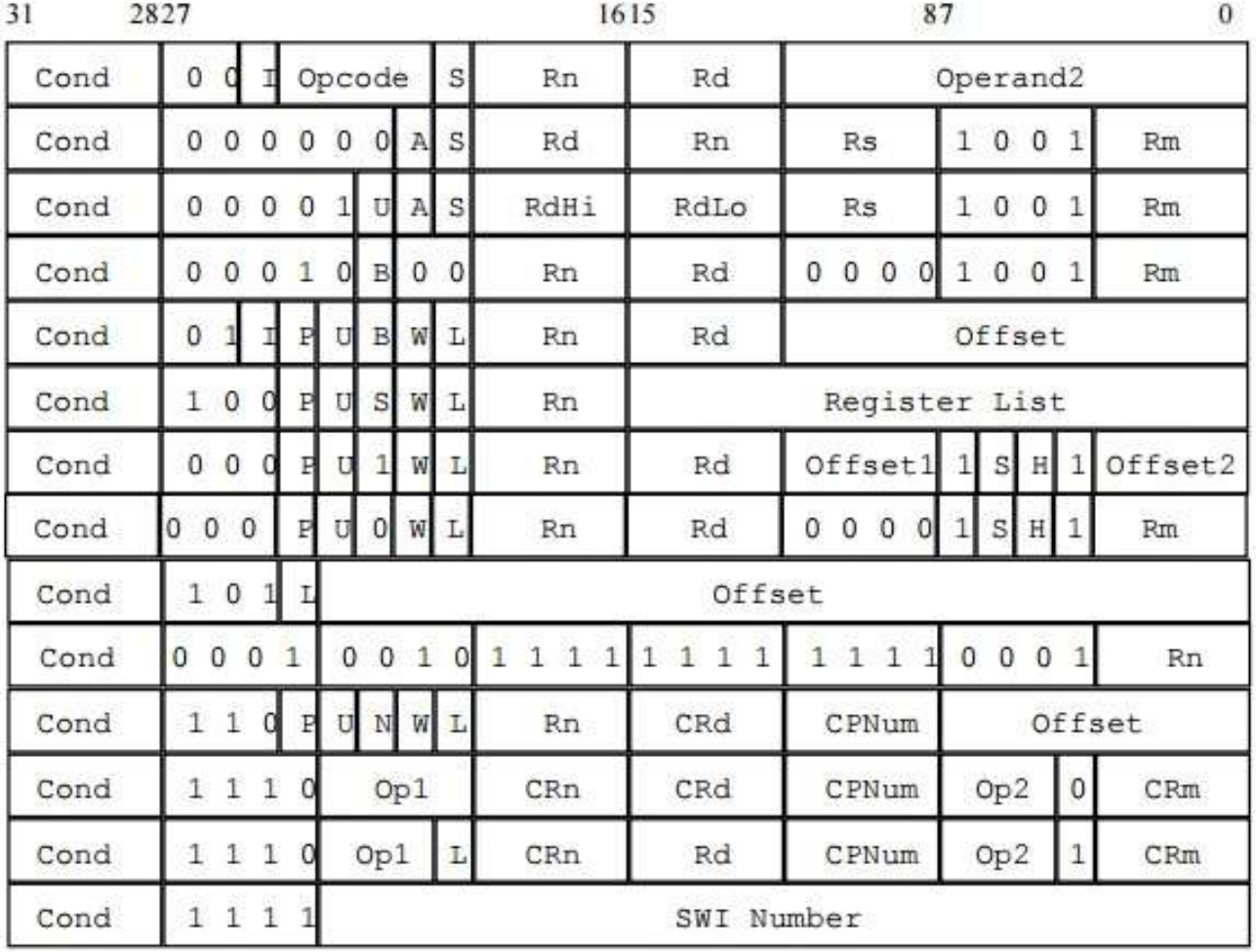
Jump: j, jal, jr, jalr



ARM

- 32-bit instructions and words
- 32 32-bit registers
- Load / Store
- 5 types of instructions

ARM Instruction Set Format



Instruction type

Data processing / PSR Transfer

Multiply

Long Multiply **(v3M / v4 only)**

Swap

Load/Store Byte/Word

Load/Store Multiple

Halfword transfer : Immediate offset **(v4 only)**

Halfword transfer: Register offset **(v4 only)**

Branch

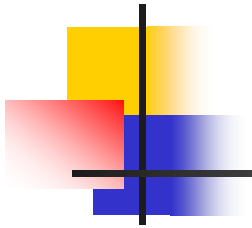
Branch Exchange **(v4T only)**

Coprocessor data transfer

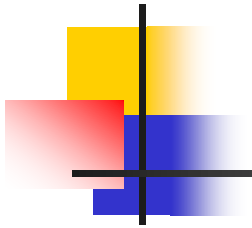
Coprocessor data operation

Coprocessor register transfer

Software interrupt



Questions?



Thank you very much.