Adaptive Compilation for an Object-Oriented, Parallel and Reconfigurable Architecture

Jecel Mattos de Assumpção Jr.
Eduardo Marques

1º Workshop de Pesquisa do LCR
June 4, 2012

Development Steps

- SiliconSqueak
- Bytecode Interpreter
- Adaptive Compilation, Partial Evaluation, Multi-level Tracing
- ALU Matrix
- Dynamic Reconfiguration
- Runtime Hardware Generation

Development Steps

This Talk

- SiliconSqueak
- Bytecode Interpreter
- Adaptive Compilation, Partial Evaluation, Multi-level Tracing
- ALU Matrix
- Dynamic Reconfiguration

SiliconSqueak

4.5 Pipeline Stages

1) Bytecode Fetch

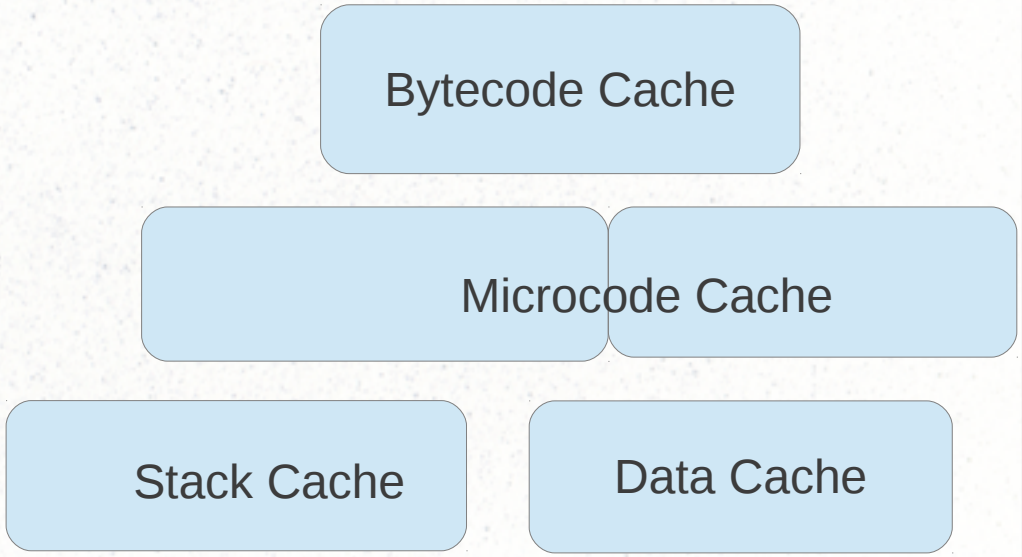
2) Microcode Fetch

3) Operand Fetch

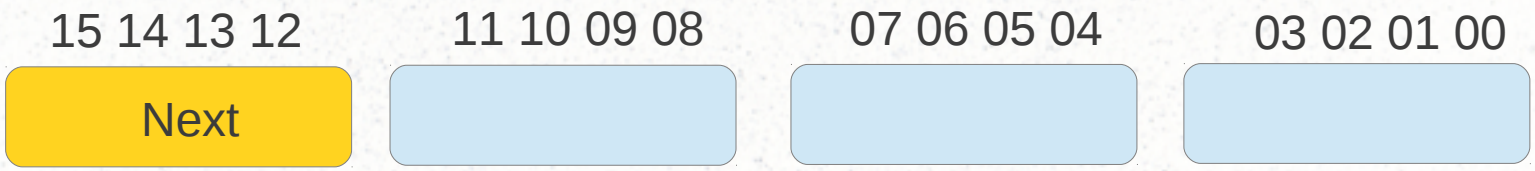
4) ALU

5) Write Back

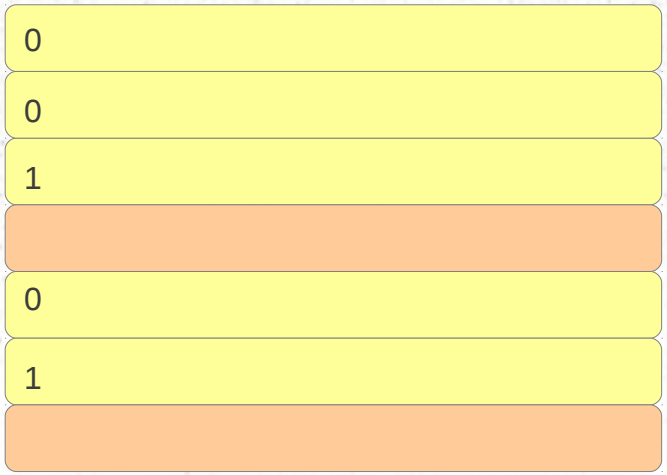
4 Caches



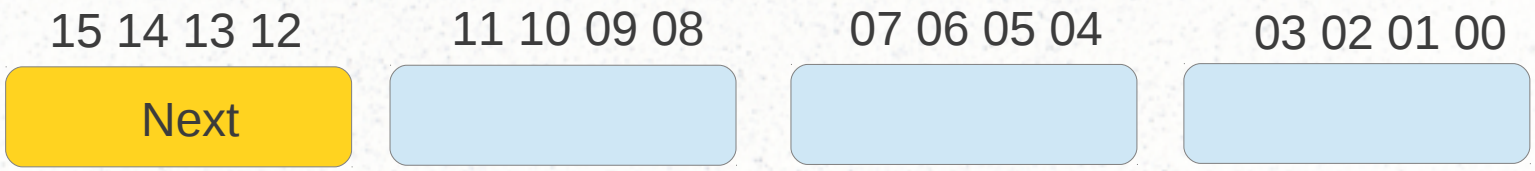
Microcode: Fetch



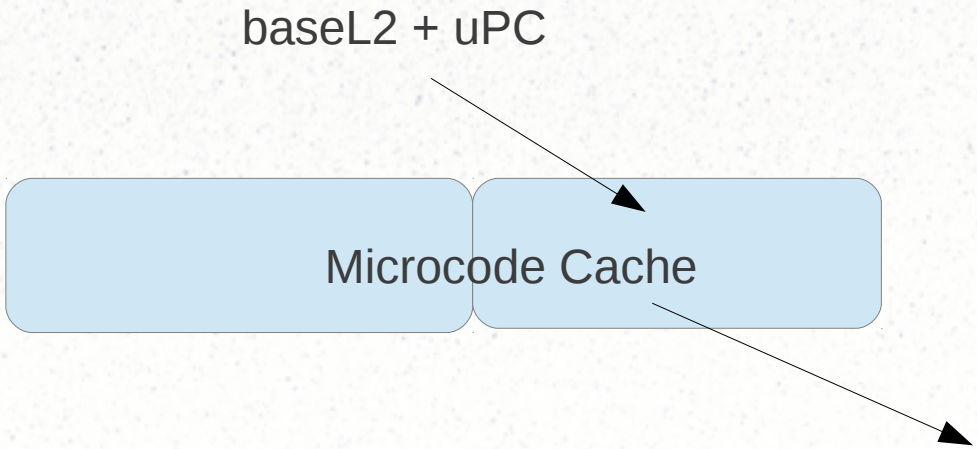
Bit 15 selects 1 or 2 word format:



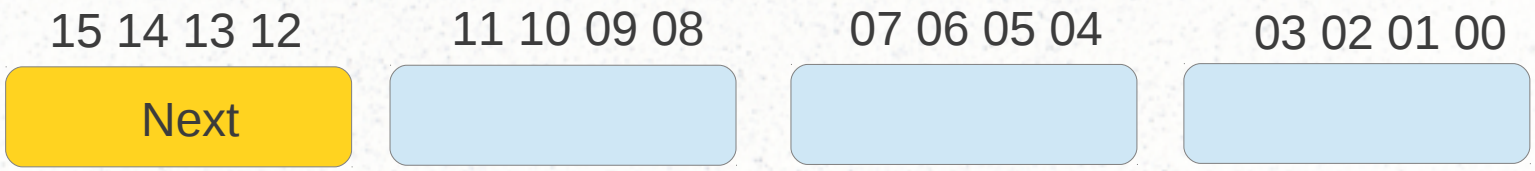
Microcode: Fetch



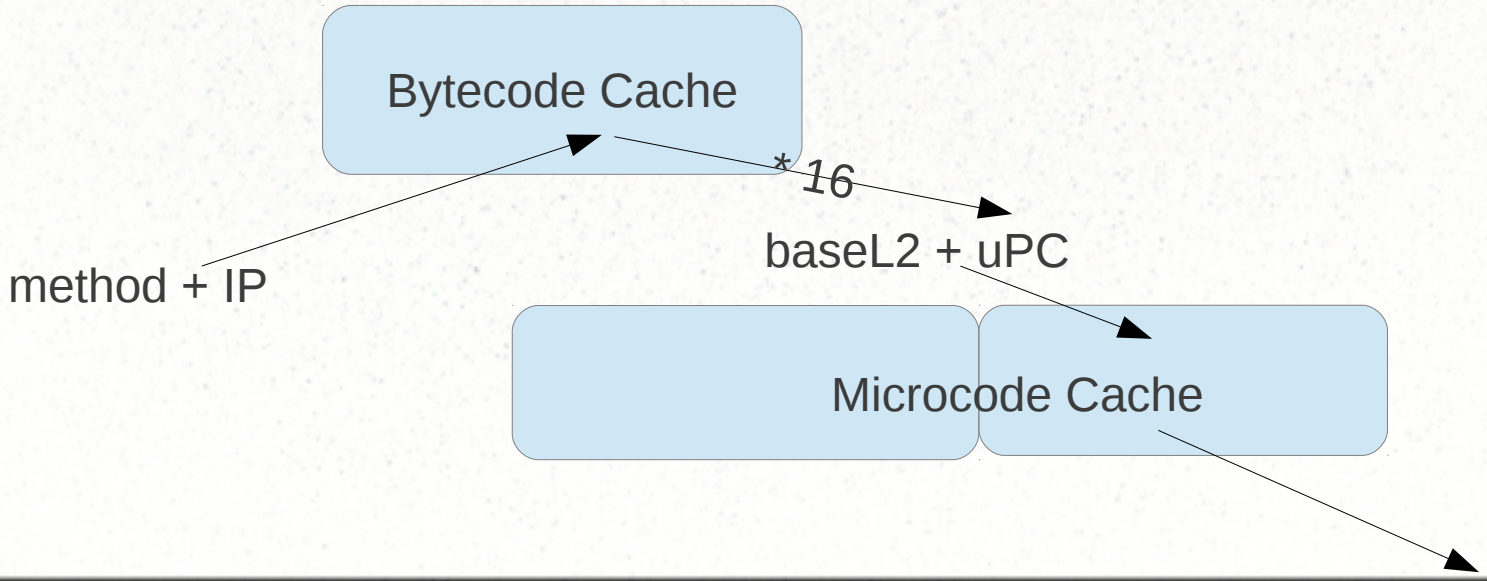
0000: Next uPC += 2



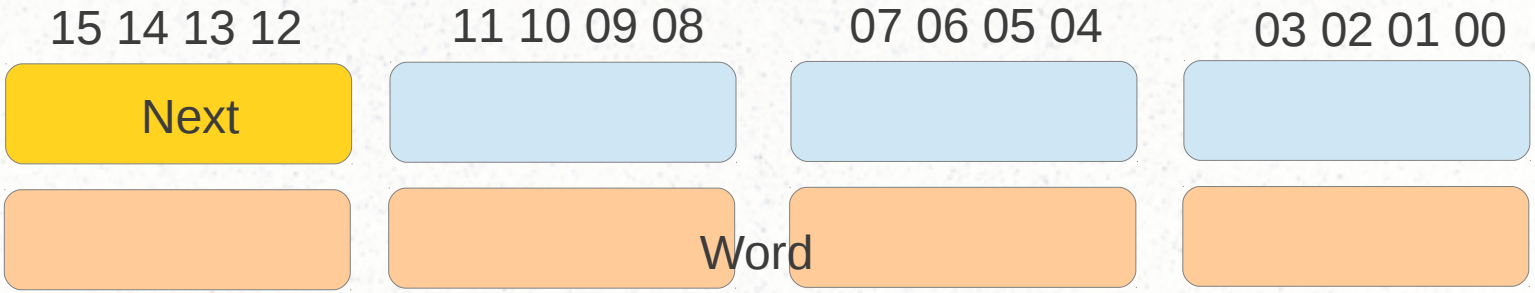
Microcode: Fetch



0110: Fetch8 uPC = byte * 16



Microcode: Fetch



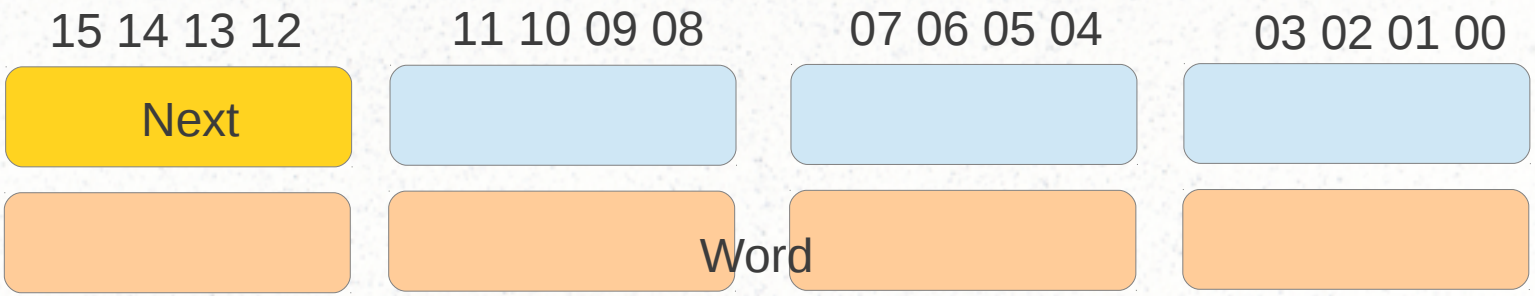
1000: Fetch1 uPC = uPC[31:16];word[15:8];byte[7];word[6:0]

1001: Fetch2 uPC = uPC[31:16];word[15:8];byte[7:6];word[5:0]

1010: Fetch3 uPC = uPC[31:16];word[15:8];byte[7:5];word[4:0]

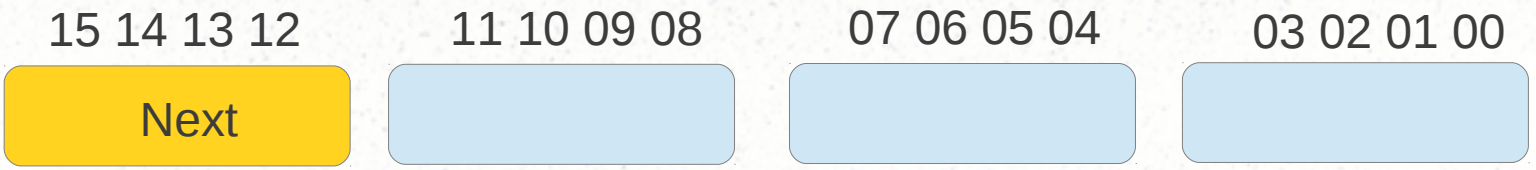
1011: Fetch4 uPC = uPC[31:16];word[15:8];byte[7:4];word[3:0]

Microcode: Fetch



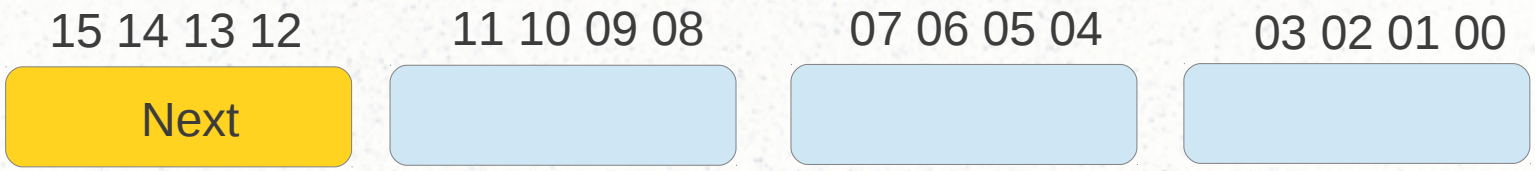
1100: Jump uPC = uPC[31:16];word[15:0]

1110: Call save uPC on return stack
uPC = uPC[31:16];word[15:0]



0001: Return uPC = load from return stack

Microcode: Fetch



0010: SkipOnZero	make next instruction NOP if result was 0
0011: SkipOnOne	make next instruction NOP if result was 1

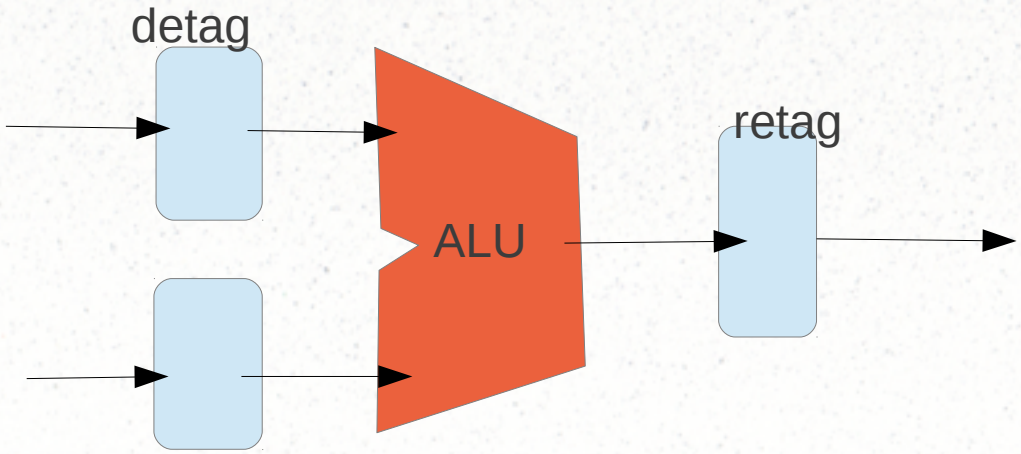
Tagged Small Integers

Squeak Small Integers:

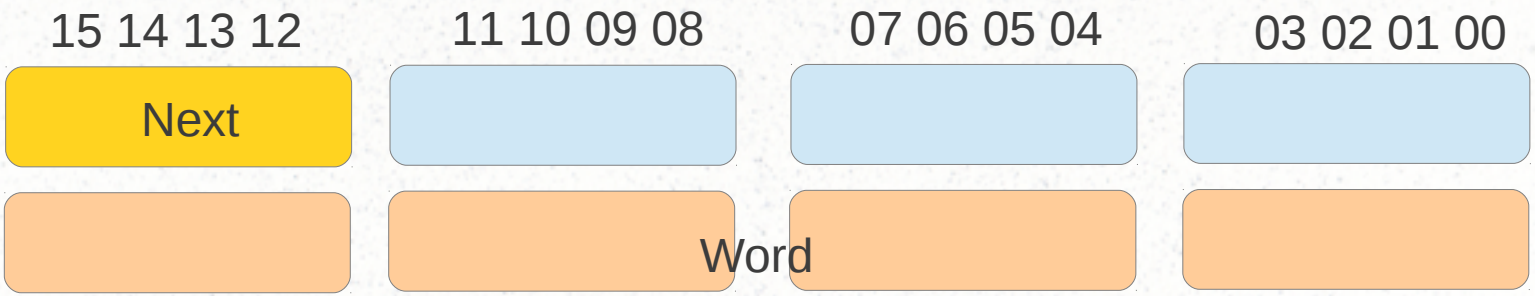
XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXX1

Squeak Object Pointers:

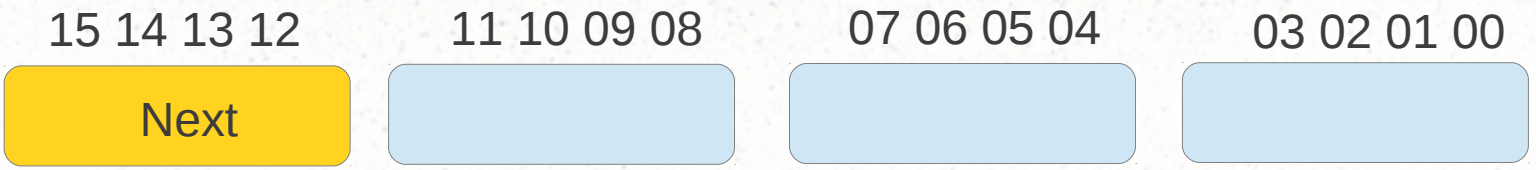
XXXX XXXX XXXX XXXX XXXX XXXX XXXX XX00



Microcode: Fetch

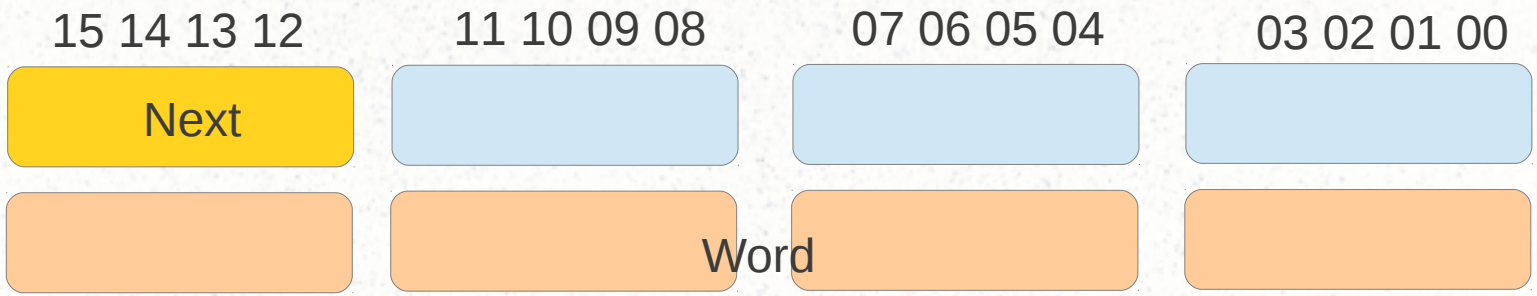


1111: TagOrCall if tag failed
save uPC on return stack
uPC = uPC[31:16];word[15:0]



0111: TagAndFetch8 if tag worked then Fetch8 else Next

Microcode: Fetch

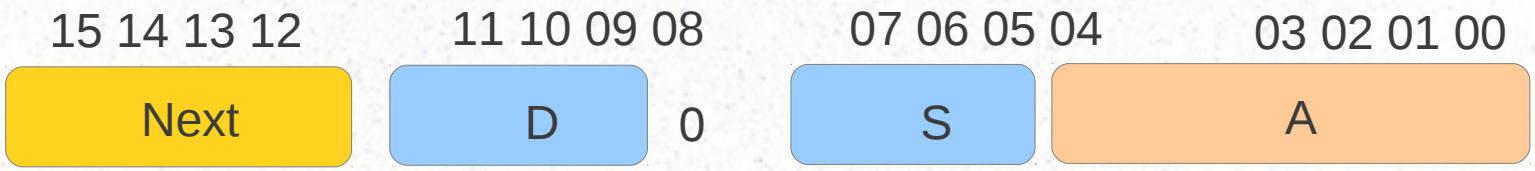


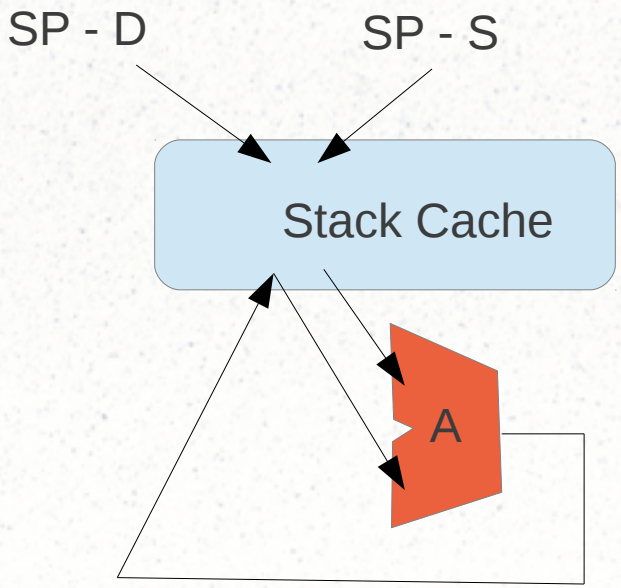
1101: PICx

0100: PICuPC

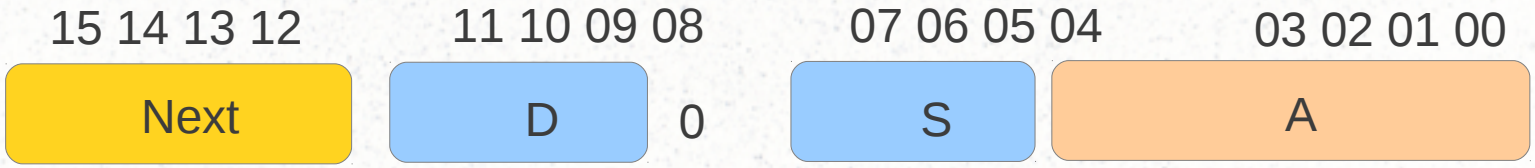
0101: PIC2

Microcode: Two Operands



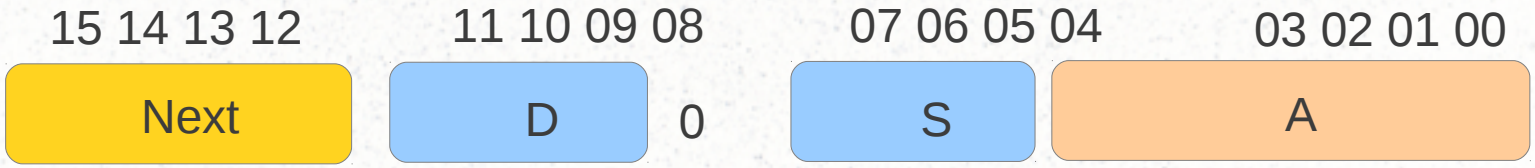


Microcode: Two Operands



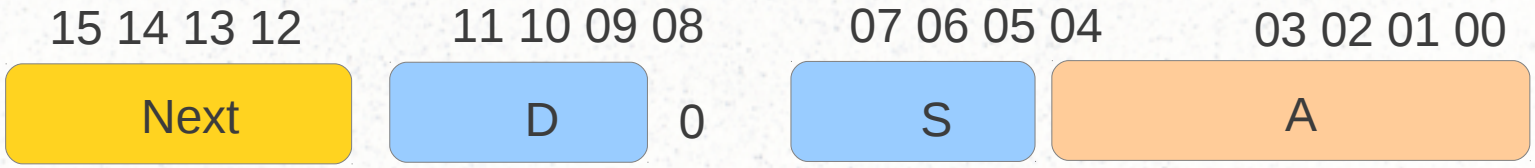
0 0000	MOV	d = s	0 0100	ADD	d = d+s
0 0001	INC	d = s+1	0 0101	?	
0 0010	NOT	d = ~s	0 0110	?	
0 0011	NEG	d = 0-s	0 0111	SUB	d = d-s

Microcode: Two Operands



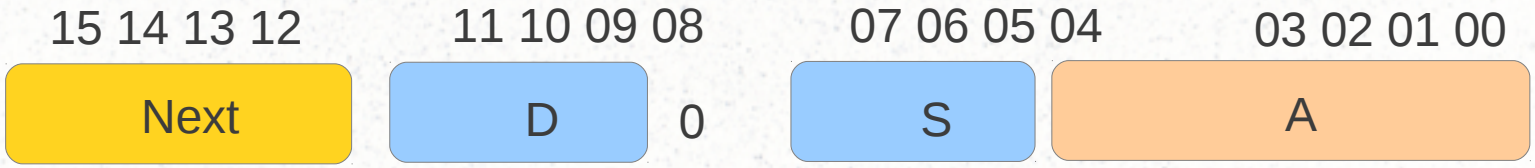
0 1000	IOR	d = d | s	0 1100	AND	d = d & s
0 1001	NOR	d = ~(d | s)	0 1101	NND	d = ~(d & s)
0 1010	XOR	d = d ^ s	0 1110	ANI	d = d & ~s
0 1011	EQV	d = ~(d ^ s)	0 1111	NAI	d = ~(d & ~s)

Microcode: Two Operands



1 0000	SHR	d = d >> s	1 0100	MUL	d = d * s
1 0001	SHL	d = d << s	1 0101	DIV	d = d / s
1 0010	ASR	d = d >> s	1 0110	UML	d = d * s
1 0011	ROT		1 0111	UDV	d = d / s

Microcode: Two Operands



1 1000	EQL	d – s == 0	1 1100	LTN	d < s
1 1001	CRY	d + s > 2**32	1 1101	LEQ	d <= s
1 1010	MIN	d – s < 0	1 1110	ULT	d < s
1 1011	OVF		1 1111	ULE	d <= s

Microcode: One Operand

K:

000	POP	U:R <= stack	100	ADR	U:R += stack
001	STR	dup; U:R <= stack	101	SBR	U:R -= stack
010	PSH	stack <= U:R	110	ADK	stack += U:R
011	INR	U:R += 1	111	SBK	stack -= U:R

Microcode: One Operand

U:

000	Temporary	stack cache	100	Literal	bytecode cache
001	Instance	data cache	101	Immediate	
010	Stream	data cache	110	ALU Matrix1	
011	Special		111	ALU Matrix2	

Squeak Bytecodes

- Push: instance variables (00-0F), temporary variables (10-1F), literals constants (20-3F), literal variables (40-5F)
- Pop: instance variables (60-67), temporary variables (68-6F)
- Jump: 90-AF
- Send Special: B0-CF
- Send Literal: No Args (D0-DF), 1 Arg (E0-EF), 2 Args(F0-FF)

Conferences (2012-2013)

deadline	when	name	where	type	qualis
December	July	ECOOP	France	OOP	A1
April	October	SPLASH	USA	OOP	A2
September	November	Smalltalks	Argentina	OOP	
June	October	SBAC-PAD	USA	Arch	B3
March	August	FPL	Norway	FPGA	B1
November	June	ISCA	USA	Arch	A1
June	August	ESUG	Belgium	OOP	
July	December	ReConFig	Mexico	FPGA	
November	March	ARC	USA	FPGA	B4
January	May	RAW		FPGA	B2

Thanks!