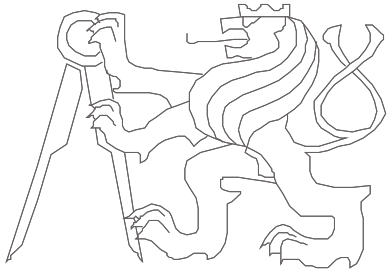
Computer Architectures

Motorola 68000, 683xx a ColdFire – CISC CPU Principles Demonstrated



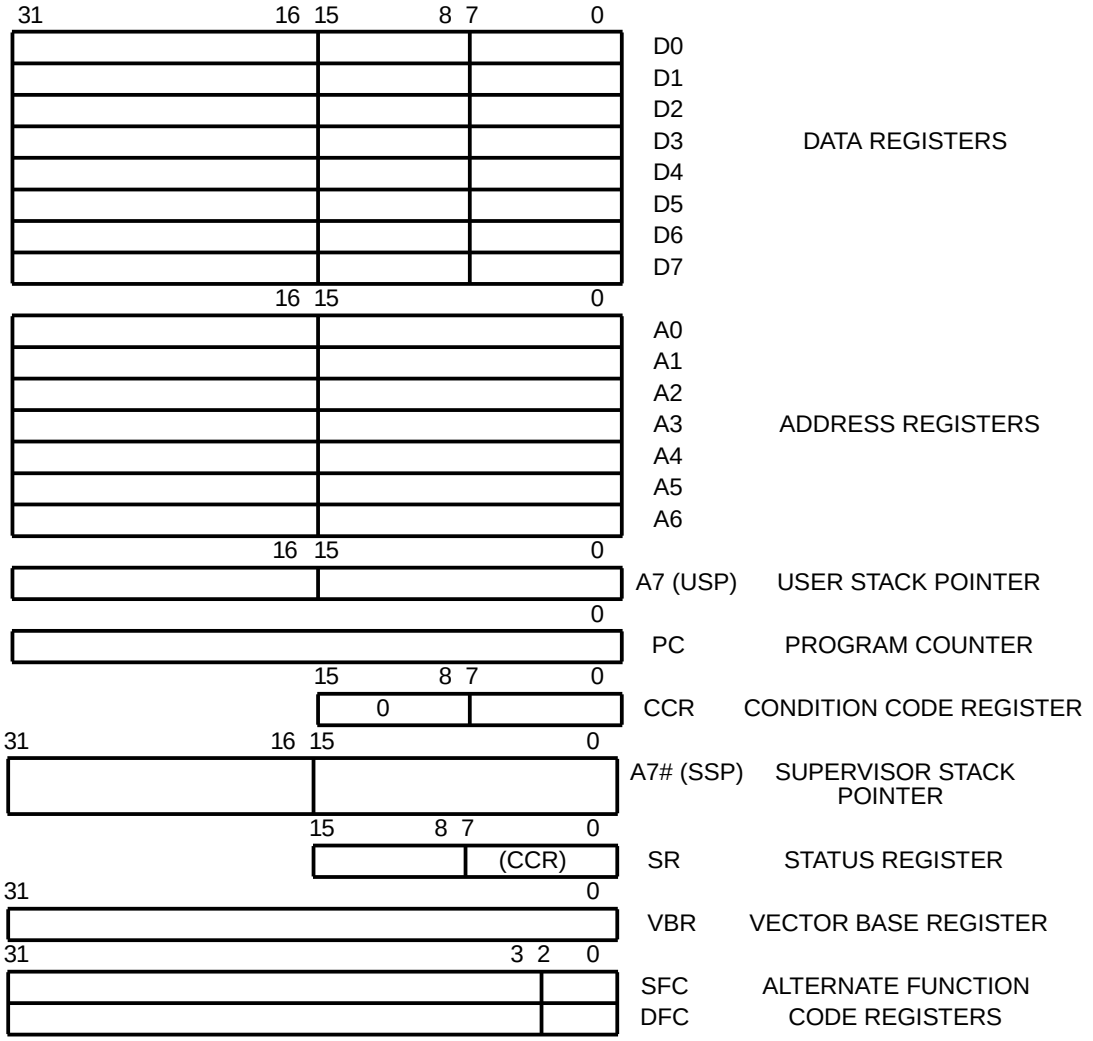
Czech Technical University in Prague, Faculty of Electrical Engineering

Original Desktop/Workstation 680X0

Feature	68000	'EC000	68010	68020	68030	68040	68060
Data bus	16	8/16	16	8/16/32	8/16/32	32	32
Addr bus	23	23	23	32	32	32	32
Misaligned Addr	-	-	-	Yes	Yes	Yes	Yes
Virtual memory	-	-	Yes	Yes	Yes	Yes	Yes
Instruct Cache	-	-	3	256	256	4096	8192
Data Cache	-	-	-	-	256	4096	8192
Memory manager	68451 or 68851			68851	Yes	Yes	Yes
ATC entries	-	-	-	-	22	64/64	64/64
FPU interface	-	-	-	68881 or 68882		Internal FPU	
built-in FPU	-	-	-	-	-	Yes	Yes
Burst Memory	-	-	-	-	Yes	Yes	Yes
Bus Cycle type	asynchronous				both	synchronous	
Data Bus Sizing	-	-	-	Yes	Yes	use 68150	
Power (watts)	1.2	0.13-0.26	0.13	1.75	2.6	4-6	3.9-4.9
at frequency of	8.0	8-16	8	16-25	16-50	25-40	50-66
MIPS/kDhryst.	1.2/2.1	2.5/4.3		6.5/11	14/23	35/60	100/300
Transistors	68k		84k	190k	273k	1,170k	2,500k
Introduction	1979		1982	1984	1987	1991	1994

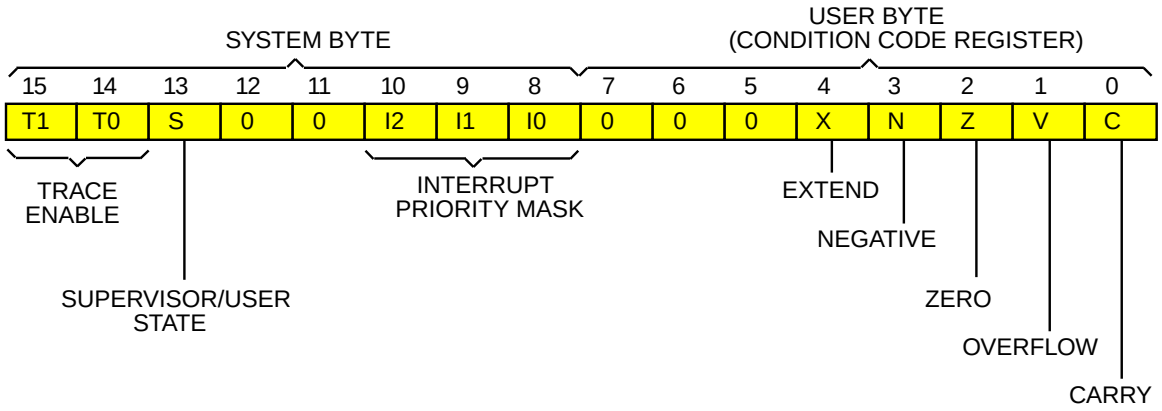
M68xxx/CPU32/ColdFire – Basic Registers Set

User programming model registers



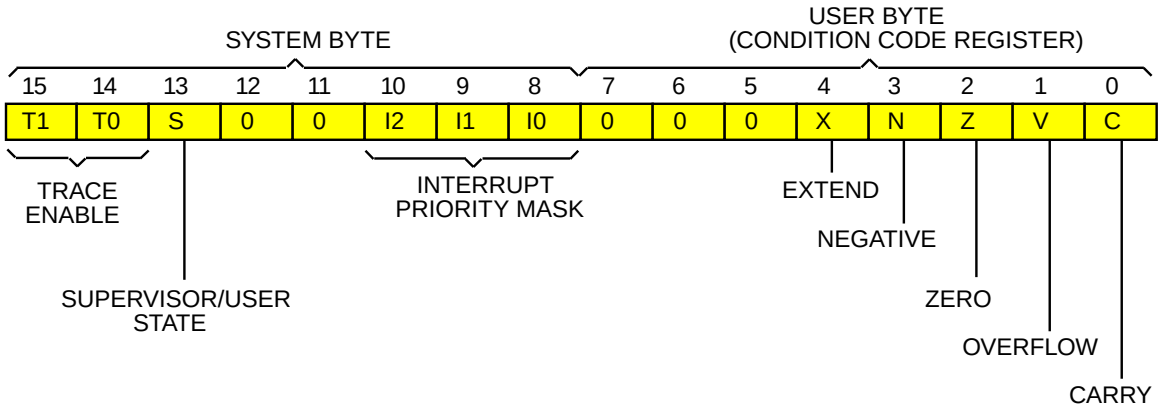
Supervisor/system programing model basic registers

Status Register – Conditional Code Part



- N – negative ... = 1 when the most significant bit of the result is set; otherwise cleared. (the result is negative for two's complement representation)
- Z – zero ... = 1 when result is zero – all bits are zero
- V – overflow .. = 1 when an arithmetic overflow occurs implying that the result cannot be represented in the operand size (signed case for add, sub, …)
- C – carry ... = 1 when when a carry out of the most significant bit occurs (add) or a borrow occurs (sub)
- X - extend (extended carry) .. Set to the value of the C-bit for arithmetic operations; otherwise not affected or set to a specified result

Status Register – System Byte



- T1, T0 – trace … if some of these bits is set then exception is generated after every instruction execution or when program flow changes (jump, call, return)
- S – supervisor … if set to 1 then CPU runs in the supervisor state/mode and SP maps to SSP. Else CPU runs in user mode, SP maps to USP and changes to the system byte are not possible and user mode privileges rules/restrictions are applied to memory access (controlled by MMU).
- I2, I1, I0 - interrupt mask … up to this interrupt priority level are requests blocked/masked – i.e. they need to wait. The level 7 is exception because it is non-maskable, i.e. exception acceptance cannot be delayed.

Addressing Modes of 68000

- Up to 14 addressing modes for operand selection exists for some architecture family members. The next described symbols convention is used for the full instructions description/specification.

Addressing – register naming convention

- **EA** - effective address
- **An** - address register n, example: **A3**
- **Dn** - data register n, example: **D3**
- **Rn** - any address or data register
- **Xn.SIZE*SCALE** index addressing with scaling
  - **Xn** any address or data register in the role of index/offset
  - **SIZE** index size **W** (16-bit) or **L** (32-bit)
  - **SCALE** - multiplier of the index 1, 2, 4 or 8
- **PC** - program counter
- **SR** - status register
- **SP** - stack pointer (**A7** - **USP** or **SSP**)
- **CCR** - conditional code register, lower byte of **SR**
- **USP** - user stack pointer
- **SSP** - supervisor stack pointer režimu

Addressing – direct operands and indirection symbolic

- **dn** offset, n bits long
- **bd** base address up to 32 bits
- **L** operand length 32 bits (long-word)
- **W** operand length 16 bits (word)
- **B** operand length 8 bits (byte)
- **(An)** brackets are used to denote that memory cell pointed by value between brackets is source/destination

Addressing modes – basic 68000 modes

- **Rn** operand represents value of data Dn or address An register
- **(An)** memory content at the address specified by **An**
- **(An)+** memory content at **An** with following **An** increment by value equivalent to the operand length (post-increment)
- **-(An)** the **An** register is decremented by operand size first and then specifies memory located operand (pre-decrement)
- **(d16,An)** memory at **An** + 16-bit sign extended offset
- **(d8,An,Xn)** memory at **An** + 8-bit sign extended offset + index register (another **Am** or D**m**) which can be eventually limited to lower 16 bits, index can be multiplied by 1, 2, 4 or 8 for CPU32 and 68020+ processors
- **(xxx).W** 16-bit absolute address – upper and lower 32kB
- **(xxx).L** 32-bit absolute address

Addressing – extended index modes and PC relative modes

- **(bd,An,Xn*SCALE)** memory address is the sum of address register, index register multiplied by **SCALE** (1, 2, 4 or 8) and up-to 32-bit base displacement (0, 16 or 32 bits), the mode encoding allows to suppress/skip index value, even address register can be suppressed, this mode is implemented on CPU32 and 68020+
- **(d16,PC)** addressing relative to **PC** with 16-bit sign extended offset
- **(d8,PC,Xn)** addressing relative to **PC** with 8-bit sign extended offset and index register (scaled) added
- **(bd,PC,Xn*SCALE)** addressing relative to **PC** with offset size up to 32 bits, index multiplied by scale; additional options are the same as for the case where address register is used as an address base

Addressing – extension found on 68020-40

The next addressing modes are listed for completeness. They are not implemented on CPU32 or ColdFire family. Only 68020 up to 68040 knows them. Even 68060 emulates them in a software.

- **([bd,An],Xn,od)** the memory operand address is computed as sum of value stored at address **An+bd,** outer displacement **od** and index
- **([bd,PC],Xn,od)** the same but **PC** relative
- **([bd,An,Xn],od)** the operand address is given by value at address **An+Xn+bd** which is then increased by up-to 32-bit outer displacement **od**
- **([bd,PC,Xn],od)** the same but **PC** relative

Instruction Format and Address Mode Encoding

SINGLE EA INSTRUCTION FORMAT

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
X	X	X	X	X	X	X	X	X	X	EFFECTIVE ADDRESS					
										MODE			REGISTER		

BRIEF FORMAT EXTENSION WORD

15	14 12	11	10 9	8	7 0
D/A	REGISTER	W/ L	SCALE	0	DISPLACEMENT

FULL FORMAT EXTENSION WORD(S)

15	14 12	11	10 9	8	7	6	5 4	3	2 0
D/A	REGISTER	W/ L	SCALE	1	BS	IS	BD SIZE	0	I/IS
BASE DISPLACEMENT (0, 1, OR 2 WORDS)									

Field	Definition	Field	Definition
Instruction Register Extension Register	General Register Number Index Register Number	BS	Base Register Suppress 0 = Base Register Added 1 = Base Register Suppressed
D/A	Index Register Type 0 = Dn 1 = An	IS	Index Suppress 0 = Evaluate and Add Index Operand 1 = Suppress Index Operand
W/L	Word/Long Word Index Size 0 = Sign-Extended Word 1 = Long Word	BD SIZE	Base Displacement Size 00 = Reserved 01 = Null Displacement 10 = Word Displacement 11 = Long-Word Displacement
Scale	Scale Factor 00 = 1 01 = 2 10 = 4 11 = 8	I/IS *	Index/Indirect Selection Indirect and Indexing Operand Determined in Conjunction with Bit 6, Index Suppress

*Memory indirect addressing will cause illegal instruction trap; must be = 000 if IS = 1

680x0 Fixed Point Instruction Set

Mnemonic	Descritption
ABCD	Add Decimal with Extend
ADD	Add
AND	Logical AND
ASL	Arithmetic Shift Left
ASR	Arithmetic Shift Right
B<*cc*>	Branch Conditionally
BCHG	Bit Test and Change
BCLR	Bit Test and Clear
BRA	Branch Always
BSET	Bit Test and Set
BSR	Branch to Subroutine
BTST	Bit Test
CHK	Check Reigster Against Bounds
CLR	Clear Operand
CMP	Compare
DB<*cc*>	Decrement and Branch Conditionally
DIVS	Signed Divide
DIVU	Unsigned Divide
EOR	Exclusive OR
EXG	Exchange Registers
EXT	Sign Extend
JMP	Jump
JSR	Jump to Subroutine
LEA	Load Effective Address
LINK	Link Stack
LSL	Logical Shift Left
LSR	Logical Shift Right

Mnemonic	Descritption
MOVE	Move Source to Destination
MULS	Signed Multiply
MULU	Unsigned Multiply
NBCD	Negate Decimal with Extened
NEG	Negate
NOP	No Operation
NOT	One's Complement
OR	Logical OR
PEA	Push effective Address
RESET	Reset External Devices
ROL	Rotate Left without Extend
ROR	Rotate Right without Extend
ROXL	Rotate Left with Extend
ROXR	Rotate Right wiht Extend
RTD	Return and Delocate
RTE	Return from Exception
RTR	Return and Restore
RTS	Return from Subroutine
SBCD	Subtract Decimal wiht Extend
S<*cc*>	Set Conditional
STOP	Stop
SUB	Subtract
SWAP	Swap data register halves
TAS	Test and Set Operand
TRAP	Trap
TRAPV	Trap on Overflow
TST	Test
UNLK	Unlink Stack Frame

Extended Instructions of CPU32 and 68020

Mnemonic	Description	CPU32	M68020
Bcc	Supports 32-Bit Displacement	yes	yes
BFxxxx	Bit Field Instructions (BFCHG, BFCLR, BFEXTS, BFEXTU, BFFO, BFINS, BFSET, BFTST)		yes
BGND	Background Operation	yes	
BKPT	New Instruction Function	yes	yes
BRA	Supports 32-Bit Displacement	yes	yes
BSR	Supports 32-Bit Displacement	yes	yes
CALLM	New Instruction		yes
CAS,CAS2	New Instruction		yes
CHK	Supports 32-Bit Operands	yes	yes
CHK2	New Instruction	yes	yes
CMP1	Supports Program Counter Relative Addressing	yes	yes
CMP2	New Instruction	yes	yes
cp	Coprocessor Instructions		yes
DIVS/DIVU	Supports 32-Bit and 64-Bit Operations	yes	yes
EXTB	Supports 8-Bit Extend to 32 Bits	yes	yes
LINK	Supports 32-Bit Displacement	yes	yes
LPSTOP	New Instruction	yes	
MOVEC	Supports New Control Registers	yes	yes
MULS/MULU	Supports 32-Bit Operands and 64-Bit Results	yes	yes
PACK	New Instruction		yes
RTM	New Instruction		yes
TBLSN,TBLUN TBLS,TBLU	New Instruction	yes	
TST	Supports Program Counter Relative, Immediate, and An Addressing	yes	yes
TRAPcc	New Instruction	yes	yes
UNPK	New Instruction		yes

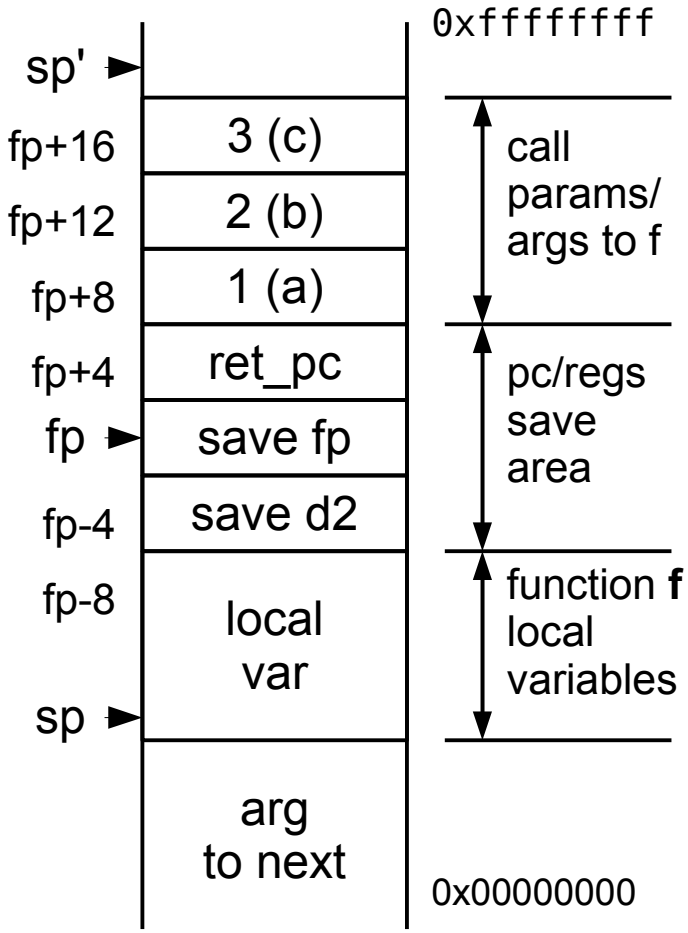
Procedure Calling and LINK/UNLK Instructions

f(1,2,3);

```
        movel #3,%sp@-
              pea 3
        movel #2,%sp@-
        movel #1,%sp@-
        jsr     f
ret_pc:
        addql #12,%sp
              lea %sp@(12),%sp
```

int f(int a, int b, int c)
{ return a + b + c; }

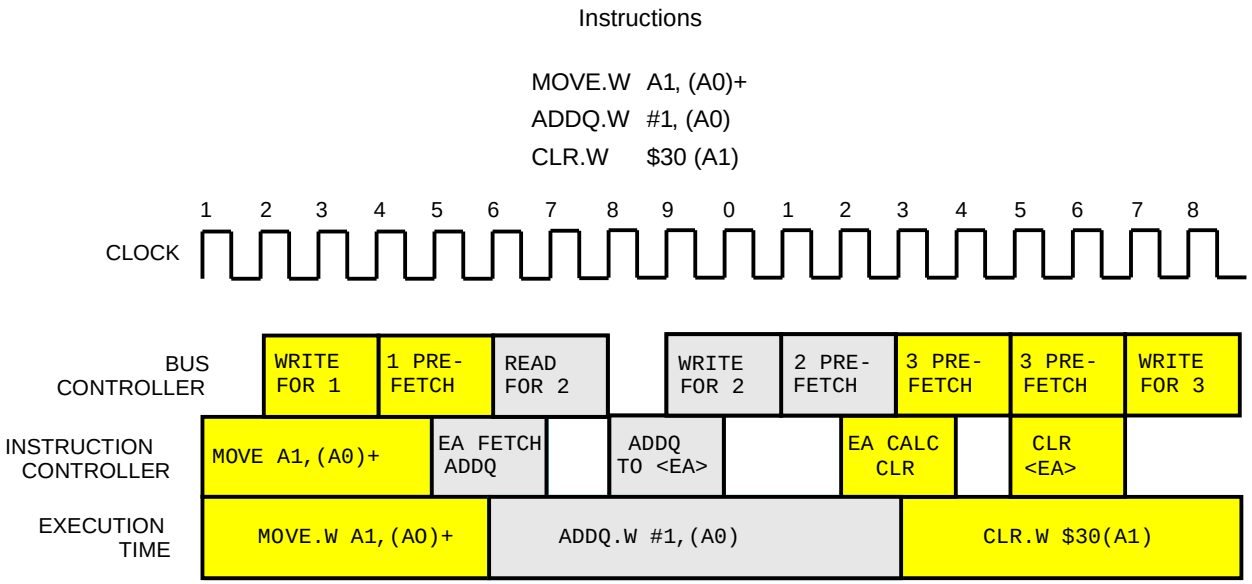
```
f:  linkw   %fp,#0
        movel %a6,%sp@-
        movel %a7,%a6
    movel %d2,%sp@-
    subql  #20,%sp

    movel %fp@(8),%d0
    addl    %fp@(12),%d0
    addl    %fp@(16),%d0

    addql  #20,%sp
    movel %sp@+,%d2
    unlk    %fp
        movel %a6,%a7
        movel %sp@+,%a6
    rts
```



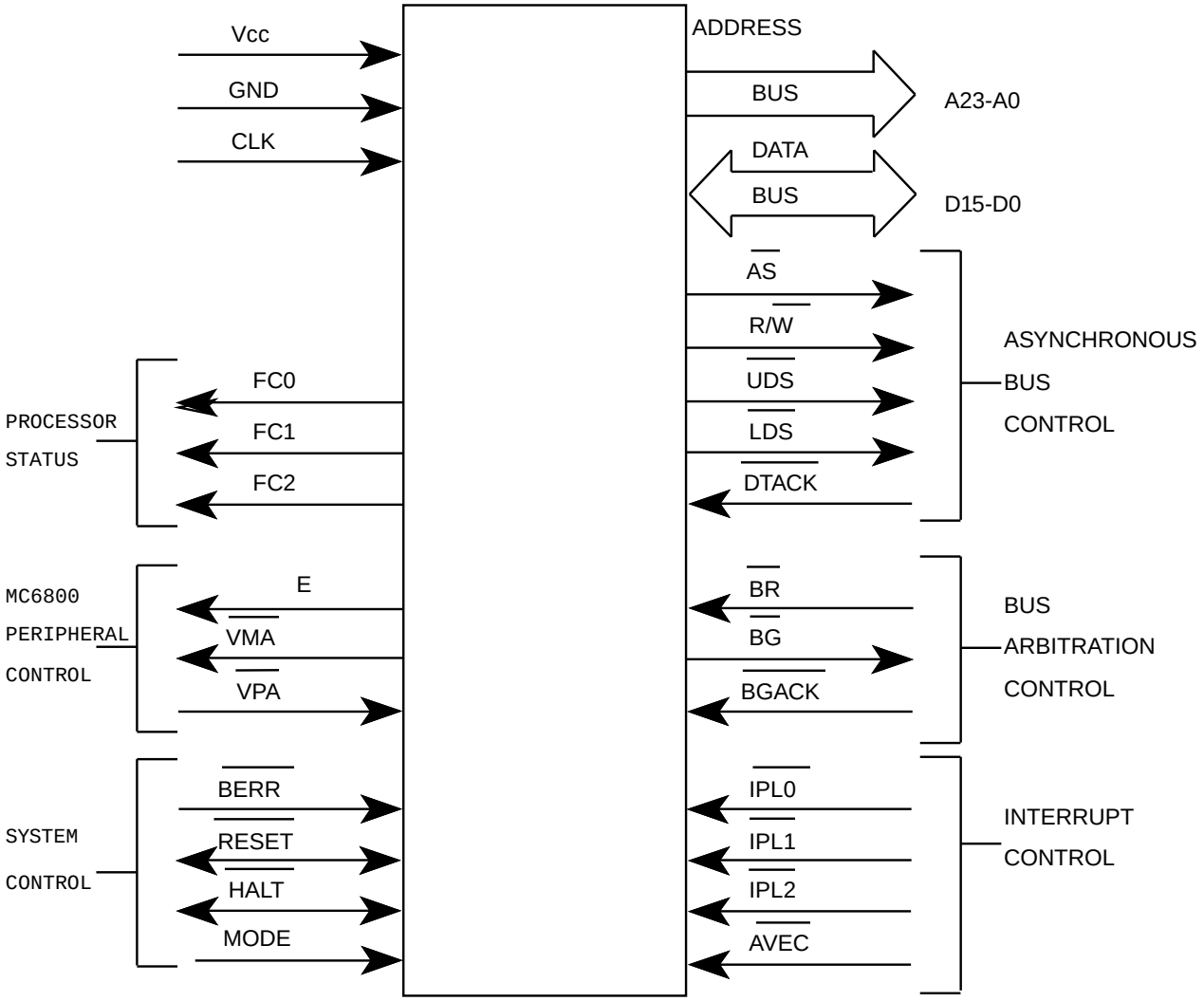
m68k calling convention

fixed A7 = SP, (A6 = FP)
return D0, D0+D1
parameters on stack
clobberable registers D0, D1, A0, A1

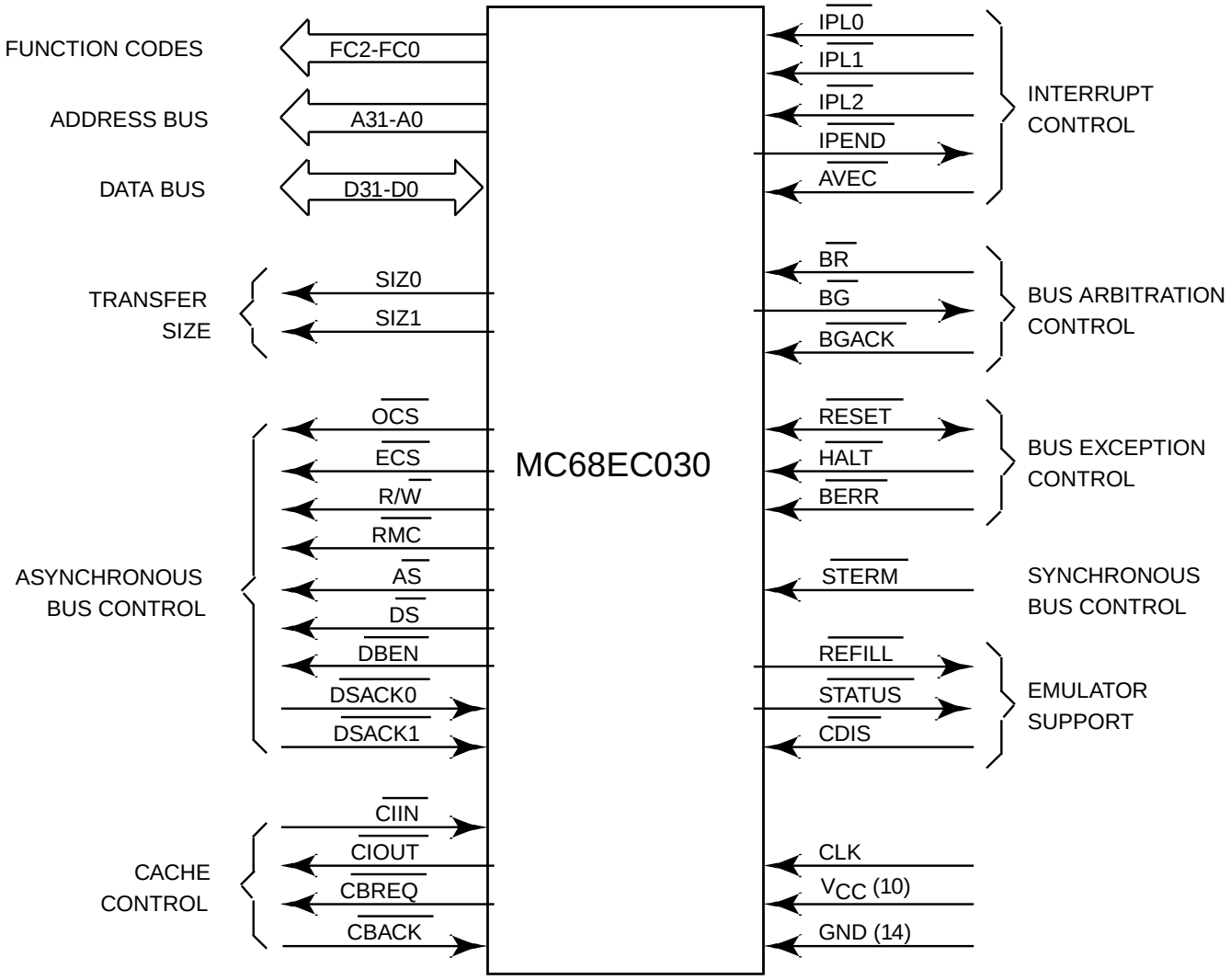
Instruction Timing for CPU32



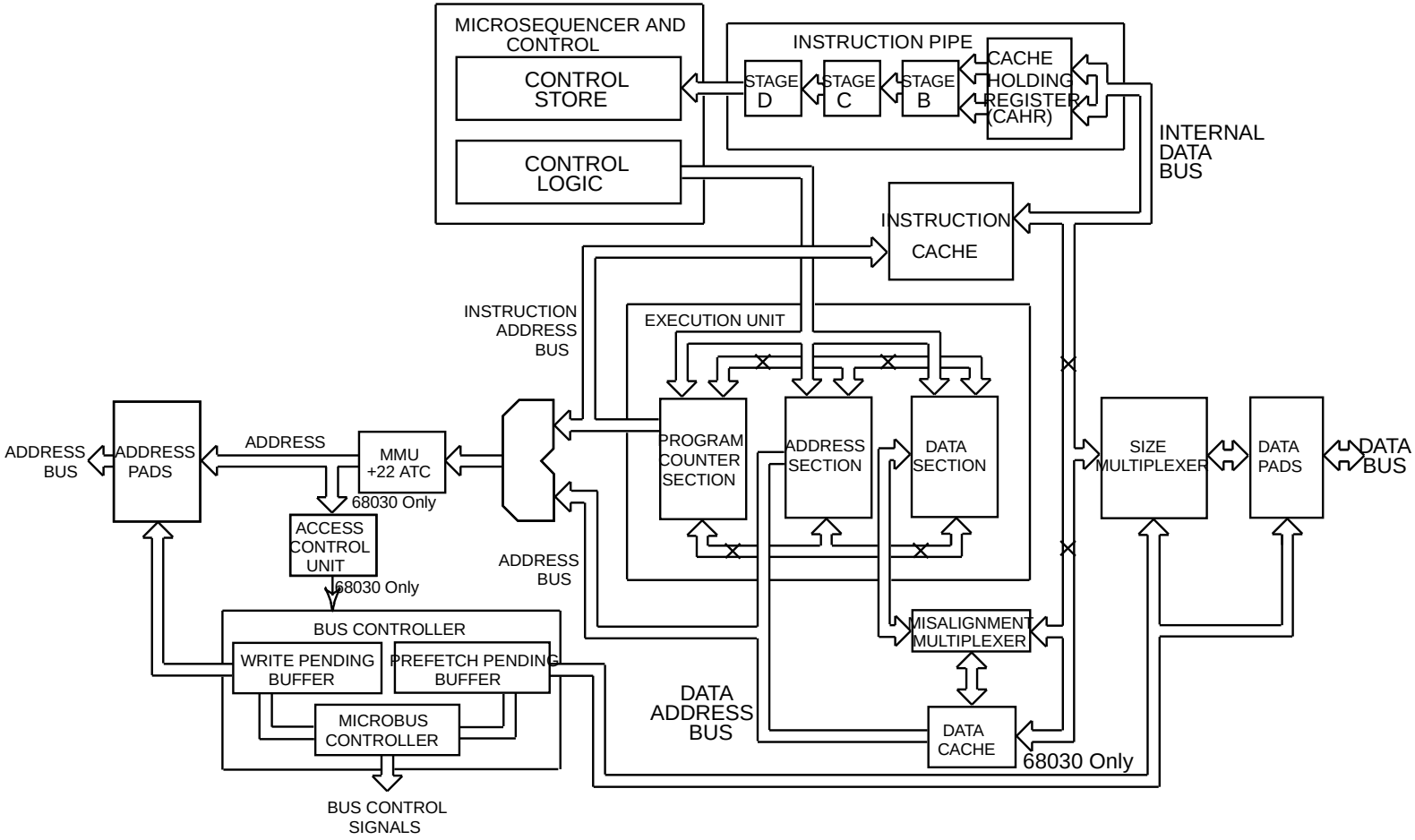
Module pads/pins/signals
68000, 68008, 68EC000



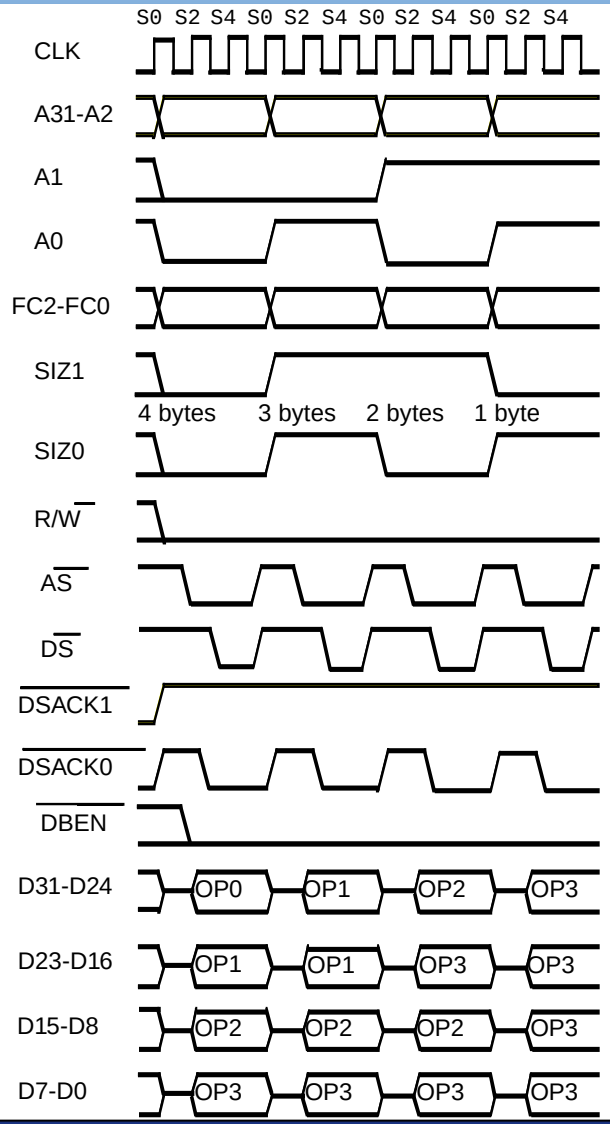
68030 Signals



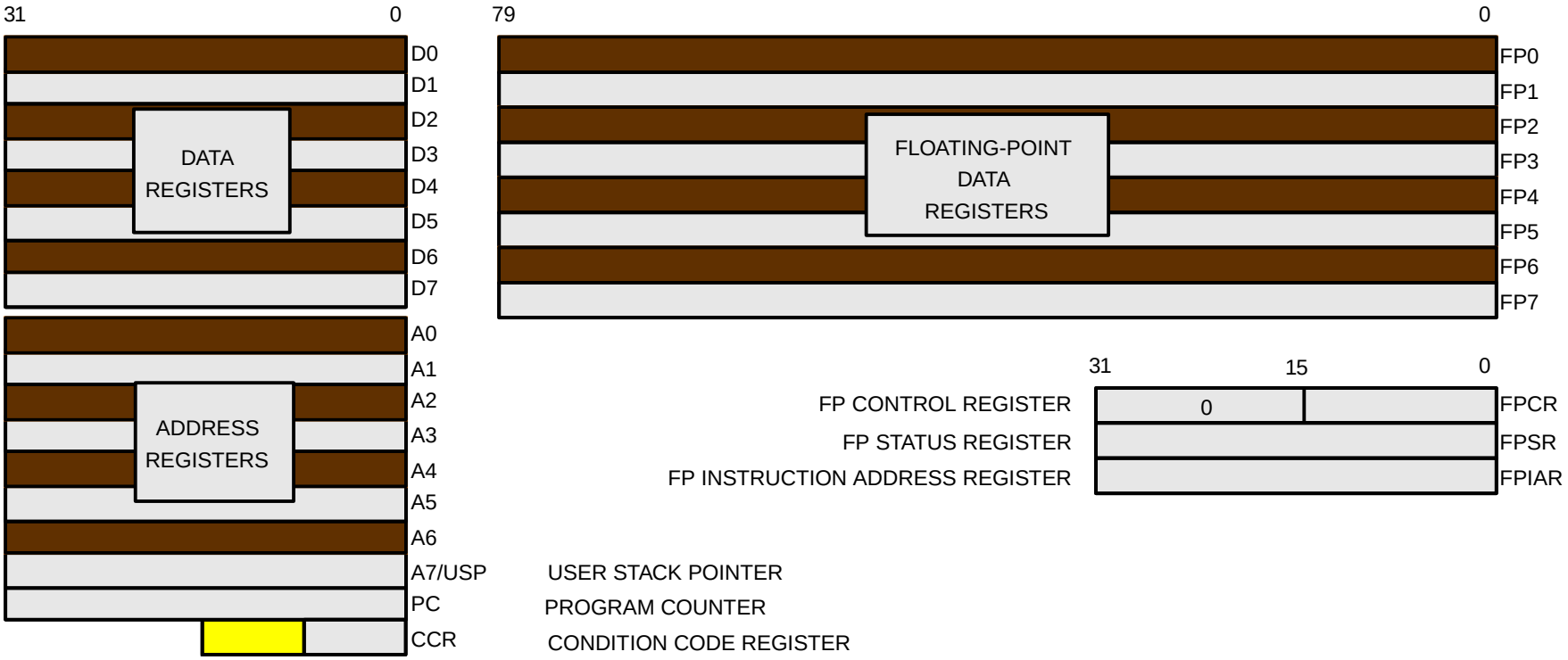
68030 Block Diagram



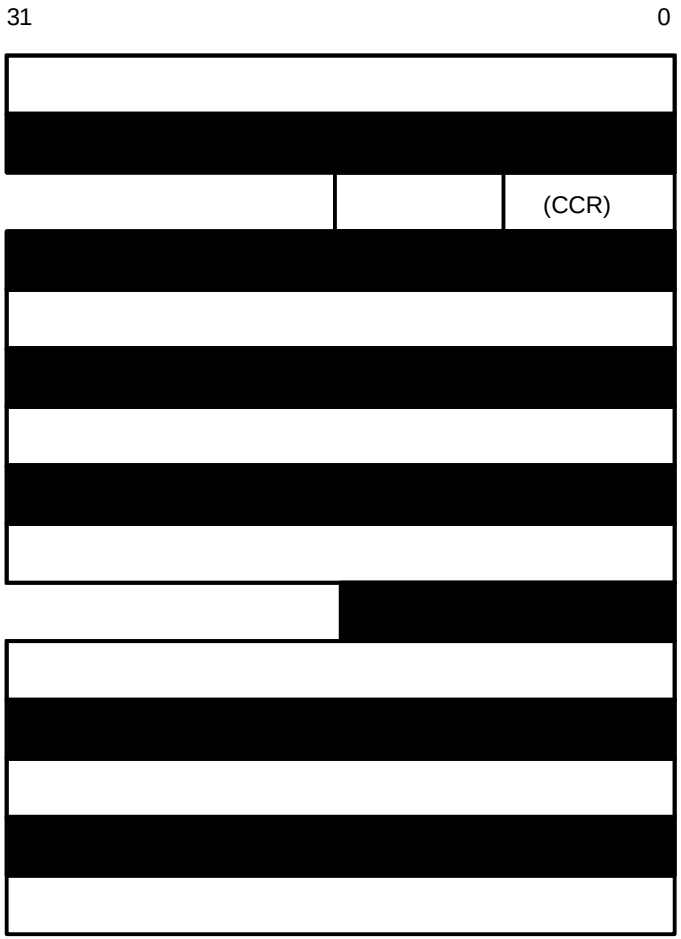
Dynamic Bus Sizing for CPU32(+) and 68020/030



68060 – User Mode



68060 – System Mode



PCR	PROCESSOR CONFIGURATION REGISTER
A7/SSP	SUPERVISOR STACK POINTER
SR	STATUS REGISTER (CCR ALSO IN THE USER PROGRAMMING MODEL)
VBR	VECTOR BASE REGISTER
SFC	SOURCE FUNCTION CODE
DFC	DESTINATION FUNCTION CODE
CACR	CACHE CONTROL REGISTER
URP	USER ROOT POINTER REGISTER
SRP	SUPERVISOR ROOT POINTER REGISTER
TC	TRANSLATION CONTROL REGISTER
DTT0	DATA TRANSPARENT TRANSLATION REGISTER 0
DTT1	DATA TRANSPARENT TRANSLATION REGISTER 1
ITT0	INSTRUCTION TRANSPARENT TRANSLATION REGISTER 0
ITT1	INSTRUCTION TRANSPARENT TRANSLATION REGISTER 1
BUSCR	BUS CONTROL REGISTER

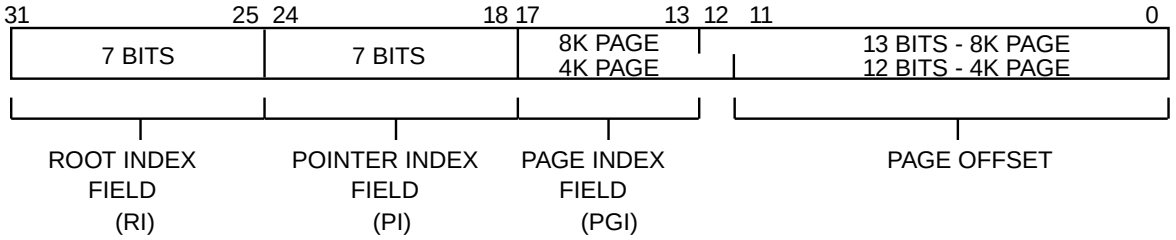
68060 – Exception vectors Assignments

Vector Number(s)	Vector Offset (Hex)	Stack Frame Format	Stacked Program Counter *	Assignment
0	000	-	-	Reset Initial SSP
1	004	-	-	Reset Initial PC
2	008	4	-	Access Fault
3	00C	2	fault	Address Error
4	010	0	fault	Illegal Instruction
5	014	2	next	Integer Divide-by-Zero
6	018	2	next	CHK, CHK2 Instructions
7	01C	2	next	TRAPcc, TRAPV Instructions
8	020	0	fault	Privilege Violation
9	024	2	next	Trace
10	028	0	fault	Line 1010 Emulator (Unimplemented A-Line Opcode)
11	02C	0	fault	Line 1111 Emulator (Unimplemented F-Line Opcode)
11	02C	2	next	Floating-Point Unimplemented Instruction
11	02C	4	next	Floating-Point Disabled
12	030	0	next	Emulator Interrupt
13	034	0	-	Only 68020, 68030 - Coprocessor Protocol Violation
14	038	0	fault	Format Error
15	03C	0	next	Uninitialized Interrupt
16-23	040-05C	-	-	(Unassigned, Reserved)
24	060	0	next	Spurious Interrupt
25	064	0	next	Level 1 Interrupt Autovector
26	068	0	next	Level 2 Interrupt Autovector
27	06C	0	next	Level 3 Interrupt Autovector
28	070	0	next	Level 4 Interrupt Autovector
29	074	0	next	Level 5 Interrupt Autovector
30	078	0	next	Level 6 Interrupt Autovector
31	07C	0	next	Level 7 Interrupt Autovector
32-47	080-0BC	0	next	TRAP #0-15 Instruction Vectors
48-55	0C0-0DC	-	-	Floating-Point Exceptions#
56	0E0	-	-	Only 68030, 68851 - PMMU Configuration
57	0E4	-	-	Only 68851 - PMMU Illegal Operation
58	0E8	-	-	Only 68851 - PMMU Access Level Violation
59	0EC	-	-	(Unassigned, Reserved)
60	0F0	0	fault	Unimplemented Effective Address
61	0F4	0	fault	Unimplemented Integer Instruction
62-63	0F8-0FC	-	-	(Unassigned, Reserved)
64-255	100-3FC	0	next	User Defined Vectors (192)

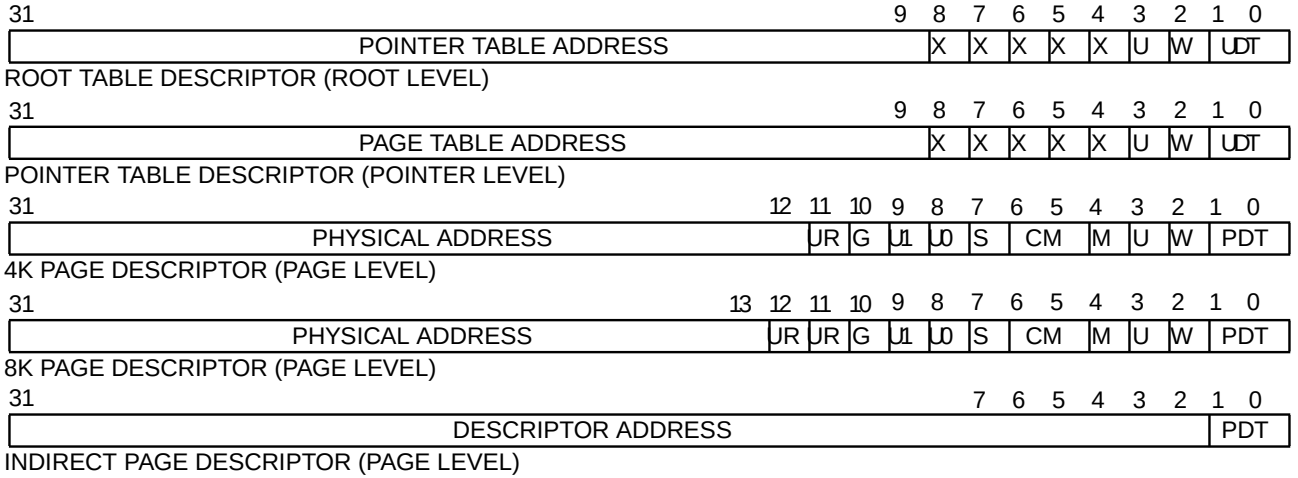
*For the Access Fault exception PC and internal CPU state necessary to finish instruction is stored "fault" refers to the PC of the instruction that caused the exception. "next" refers to the PC of the next instruction that follows the instruction that caused the fault.

68060 – Paging

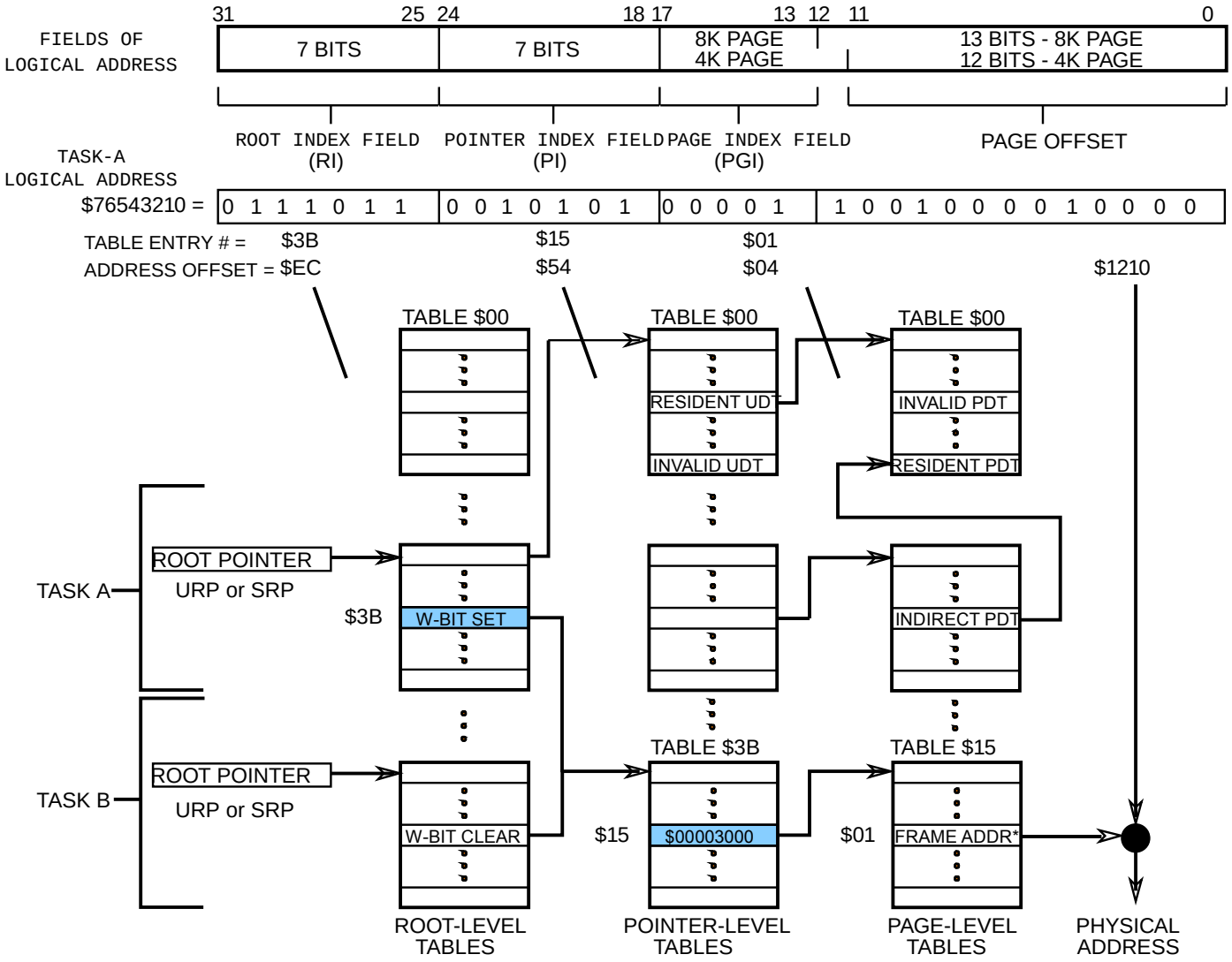
Virtual address format and bit mapping to page table levels



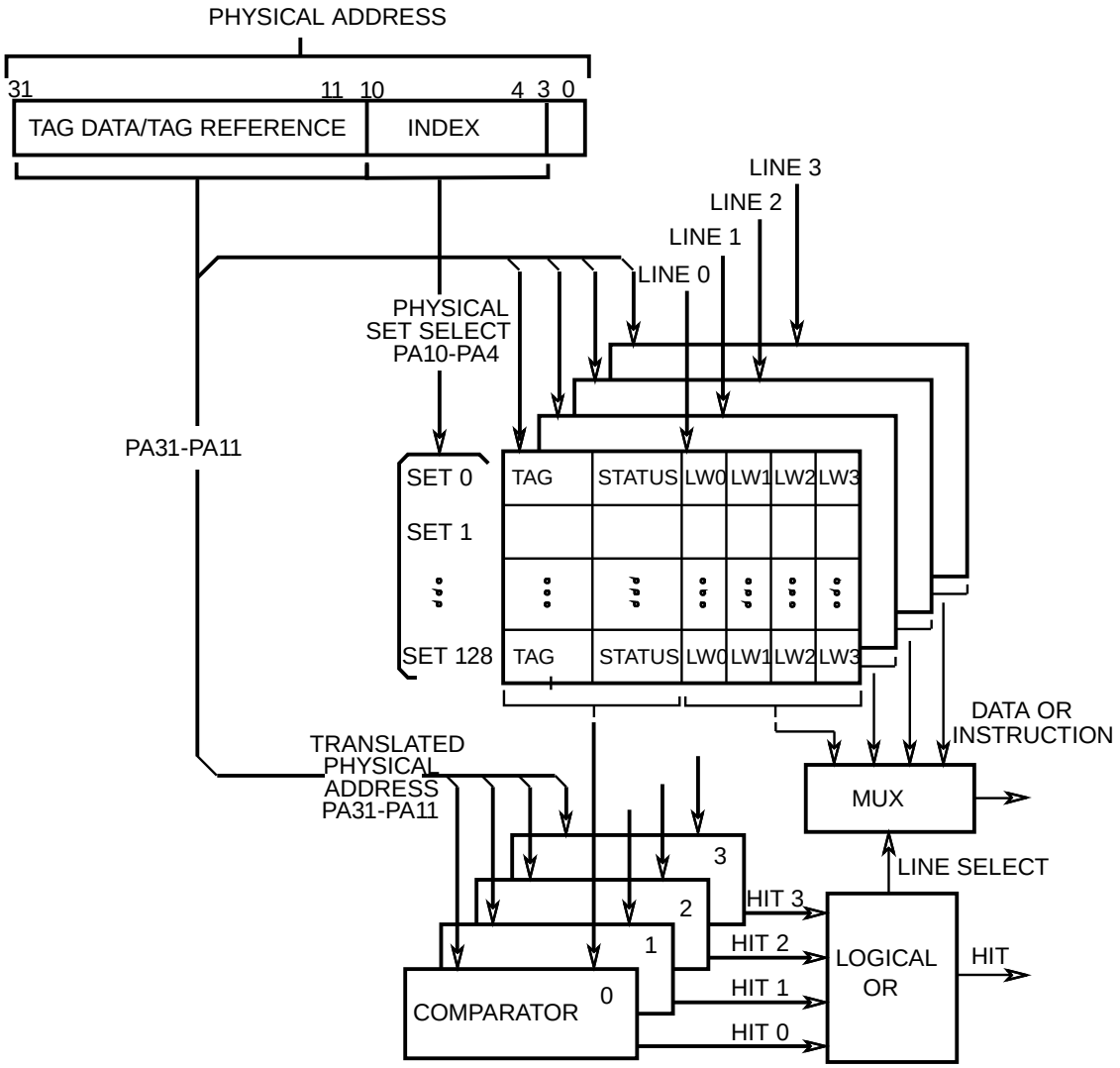
Page table descriptors



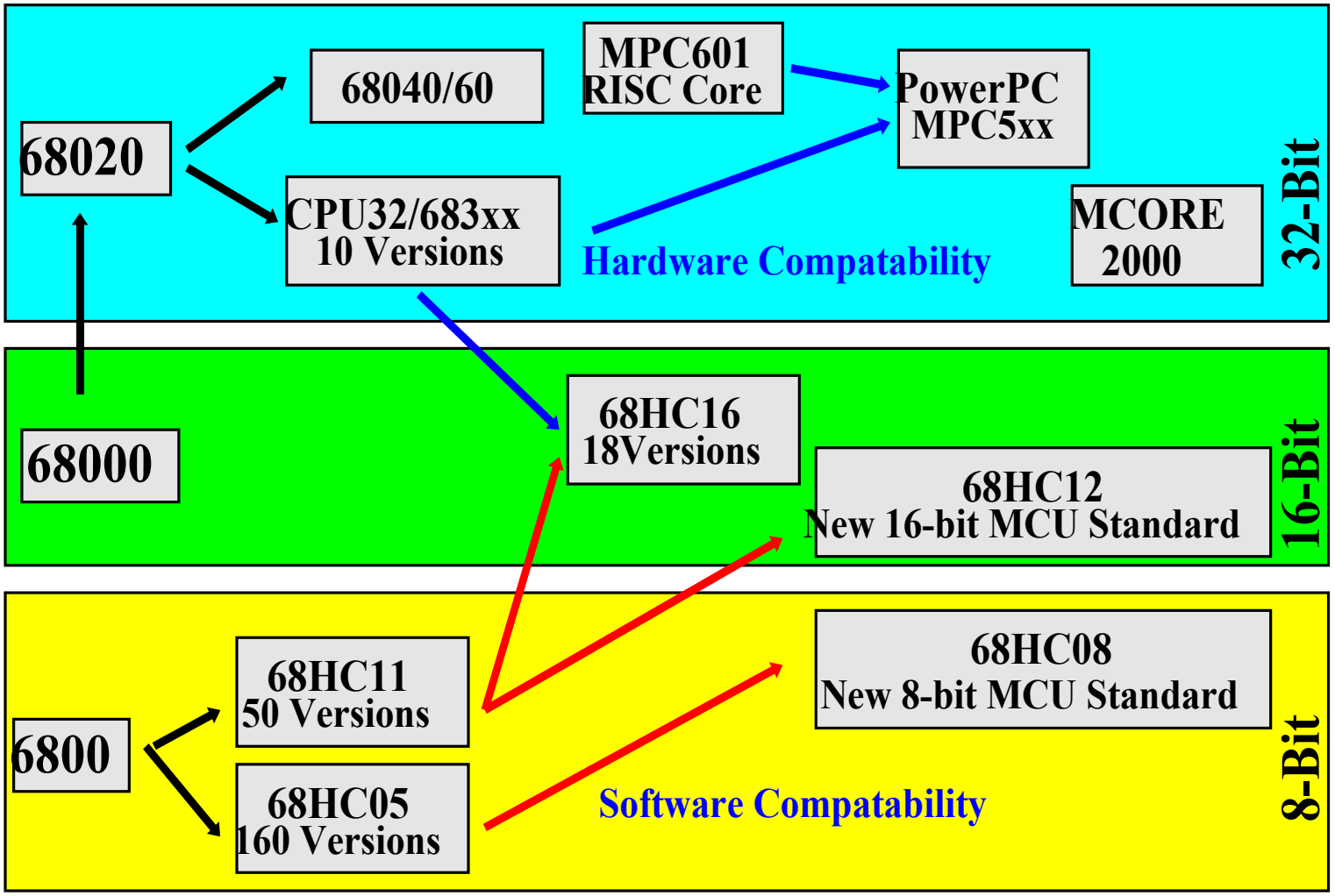
68060 – Page Table and Address Translation



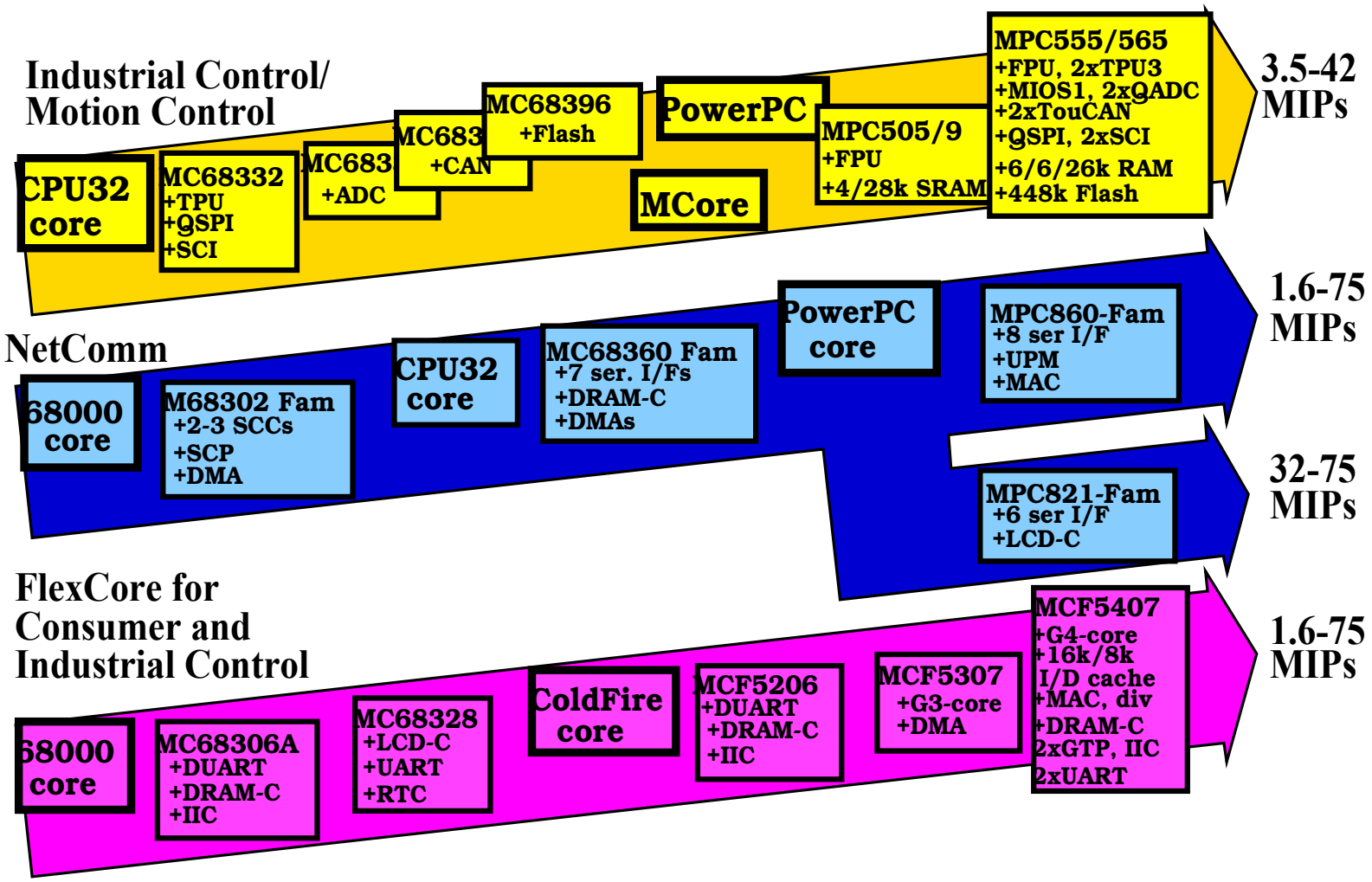
68060 – Instruction/Data Cache Memory Organization



The Evolution of Motorola/FreeScale's CPUs and MCUs



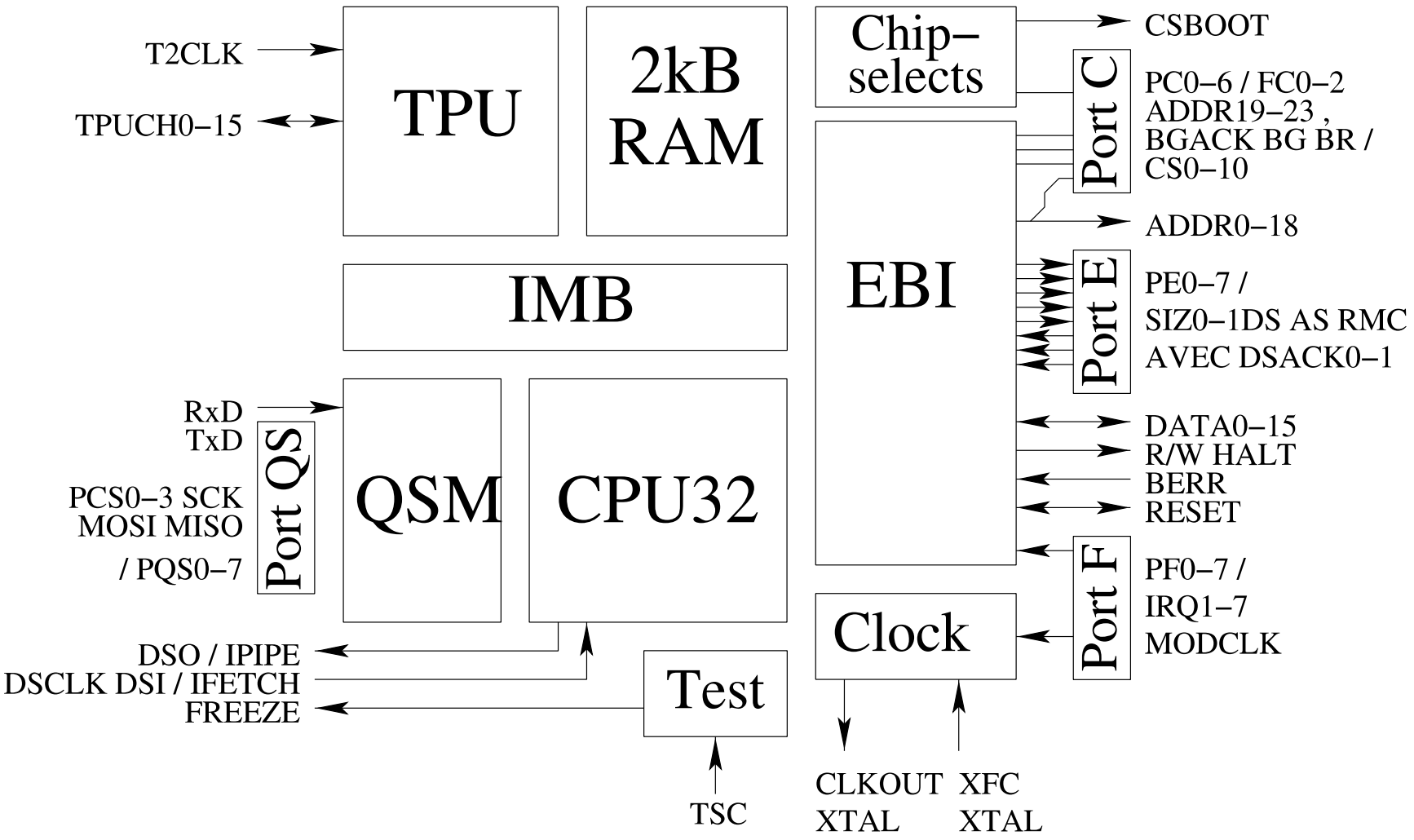
32-bit MCUs and CPUs for embedded applications



683xx/CPU32(+) Microcontroller Family

Feature	68332	68376	68360	68VZ328
Core CPU	CPU32	CPU32	CPU32+	FLX68000
Data Bus	8/16	8/16	8/16/32	8/16
Addr Bus	24	24	32	24/32MB DRAM
Misaligned Addr	-	-	Yes	
Development Int.	BDM	BDM	BDM/JTAG	ICE
TPU (timer)	Yes	Yes		
UART			2xSMC	2x
DRAM controller			Yes	EDO, FP, SD
Static Ram	2K	3.5K+4K	2.5K	
Flash EEPROM				
A/D Converter		8/10 bits		
Serial Ports	1xSCM	1xSCM	4xSCC	
SPI interface	1xQSM	1xQSM	1xSCP	2x
DMA			2 ch	
Timer		CTM4 (8)	4x16, 2x32	2x+2xPWM
Parallel Ports (bits)	up 4 (31)	up 6 (47)	3	10 (78)
Chip Selects X	12	12	8	8
More ...		TouCAN	opt. Ethernet	LCD, RTC
Clock speed Mhz	16/20/25	16/20/25	25/33	up 33
Power voltage	5V	5V	3.3 or 5V	2.7-3.3
Power (watts)	0.6	0.6	0.3-1.0	0.06-0.1
at frequency of	20	20.97	25	33

68332 Microcontroller



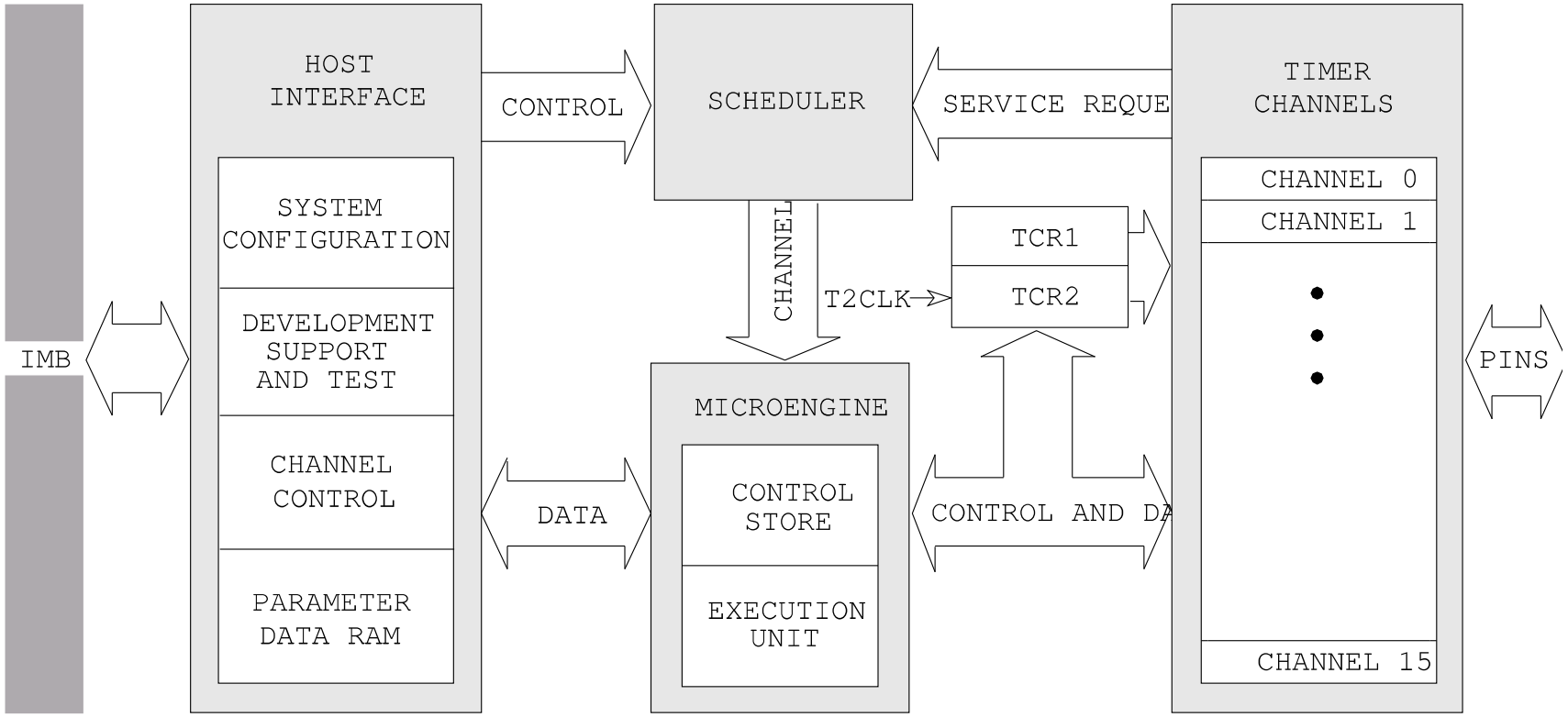
System Integration Module (SIM)

- Interconnects intermodule buss and external bus
- Includes support for programmable chip selects generators
- Ensures system protection/recovery from error states
- SIM provides software watchdog monitor and a periodic interrupt timer to support execution of time-critical control routines.
- It monitors system clock source frequency and system bus.
- The system clock can be based on 32.768-kHz crystal to achieve low power consumption.
- The module includes factory test and debugging hardware

Central Processing Unit (CPU)

- 32-bit architecture
- Compatible with MC68000 and MC68010 processors
- New instructions for embedded and control applications
- Virtual memory support only with external MMU (Memory Management Unit)
- Accelerated processing of the loop including single instruction
- Instructions for table processing and interpolation
- Exceptions processing extended to support embedded applications
- Provides tracing support with program flow change detection (return, subroutine call, ...)
- External HW breakpoint input and complete support for in system debugging – Background Debug Mode
- Fully static implementation allows clock frequency decrease and even CPU clock stopping

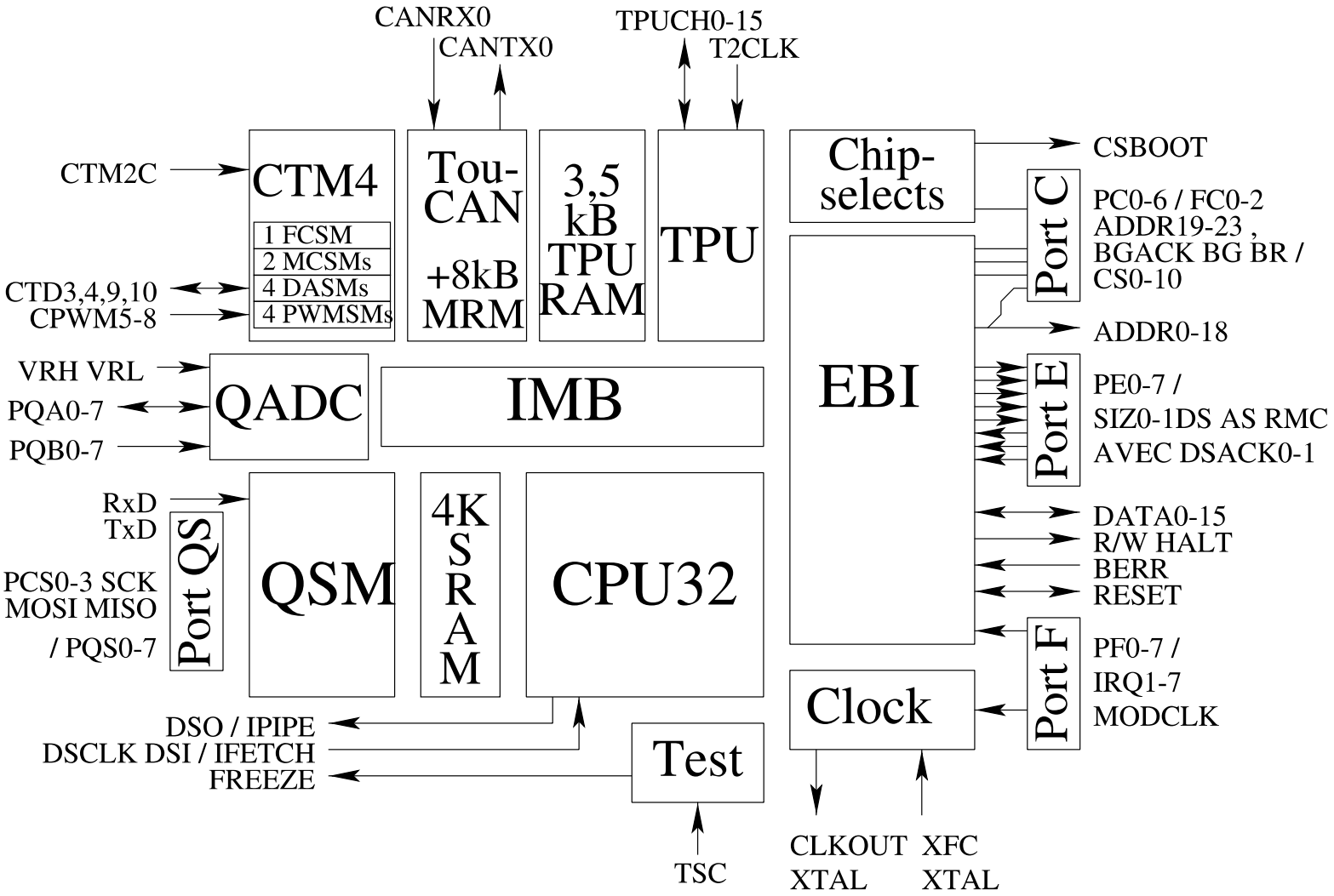
Time Processor Unit (TPU)



Time Processor Unit (TPU)

- Dedicated micro-engine operating independently of the CPU32
- 16 independent programmable channels and pins
- Each channel has an event register consisting of a 16-bit capture register, a 16-bit compare register and a 16-bit comparator
- Any channel can perform any time function
- Each channel has six or eight 16-bit parameter registers
- Each timer function may be assigned to more than one channel
- Two timer counter registers with programmable prescalers
- Each channel can be synchronized to one or both counters
- Selectable channel priority levels

68376 Microcontroller



Questions?