АРХИТЕКТУРА СОВРЕМЕННЫХ ЭВМ

Лекция 5:
Уровень архитектуры набора команд

ВМК МГУ им. М.В. Ломоносова, Кафедра АСВК
Доцент, к.ф.-м.н. Волканов Д.Ю.

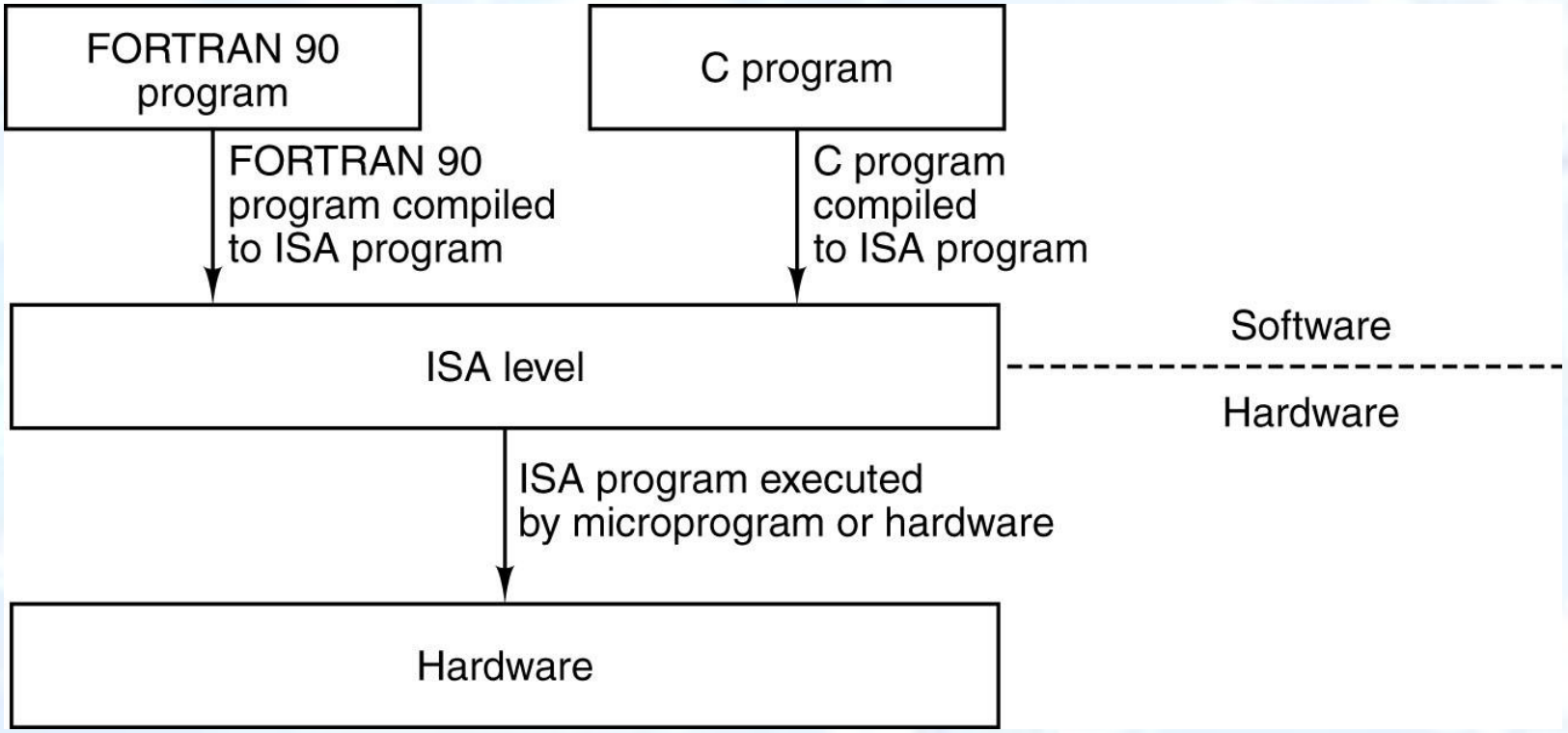
План лекции

- Регистры
- Виды адресации
- Примеры команд
- Прерывания

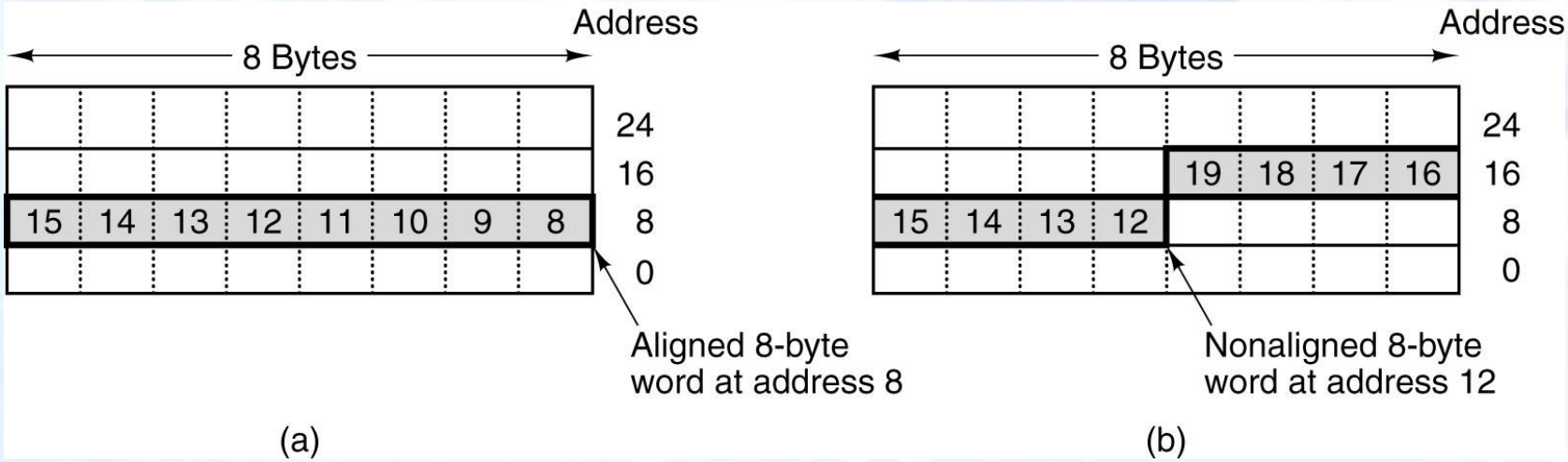
Уровни архитектуры

- Цифровой логический уровень
- Уровень микроархитектуры
- **Уровень архитектуры набора команд**
- Уровень операционной системы
- Уровень ассемблера

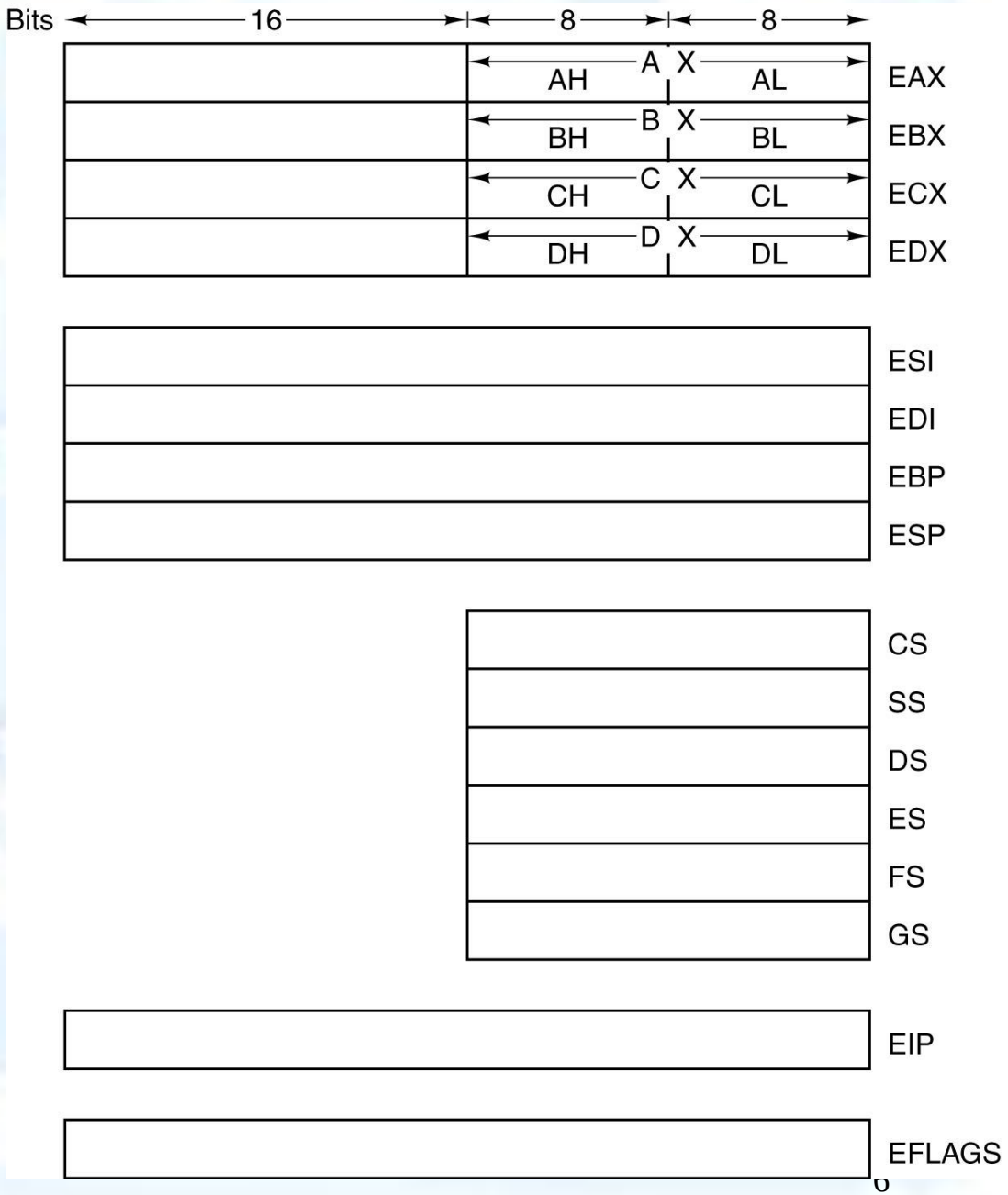
Уровень архитектуры набора команд

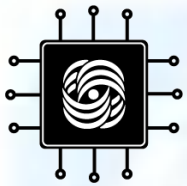


Модели памяти



Регистры Pentium 4

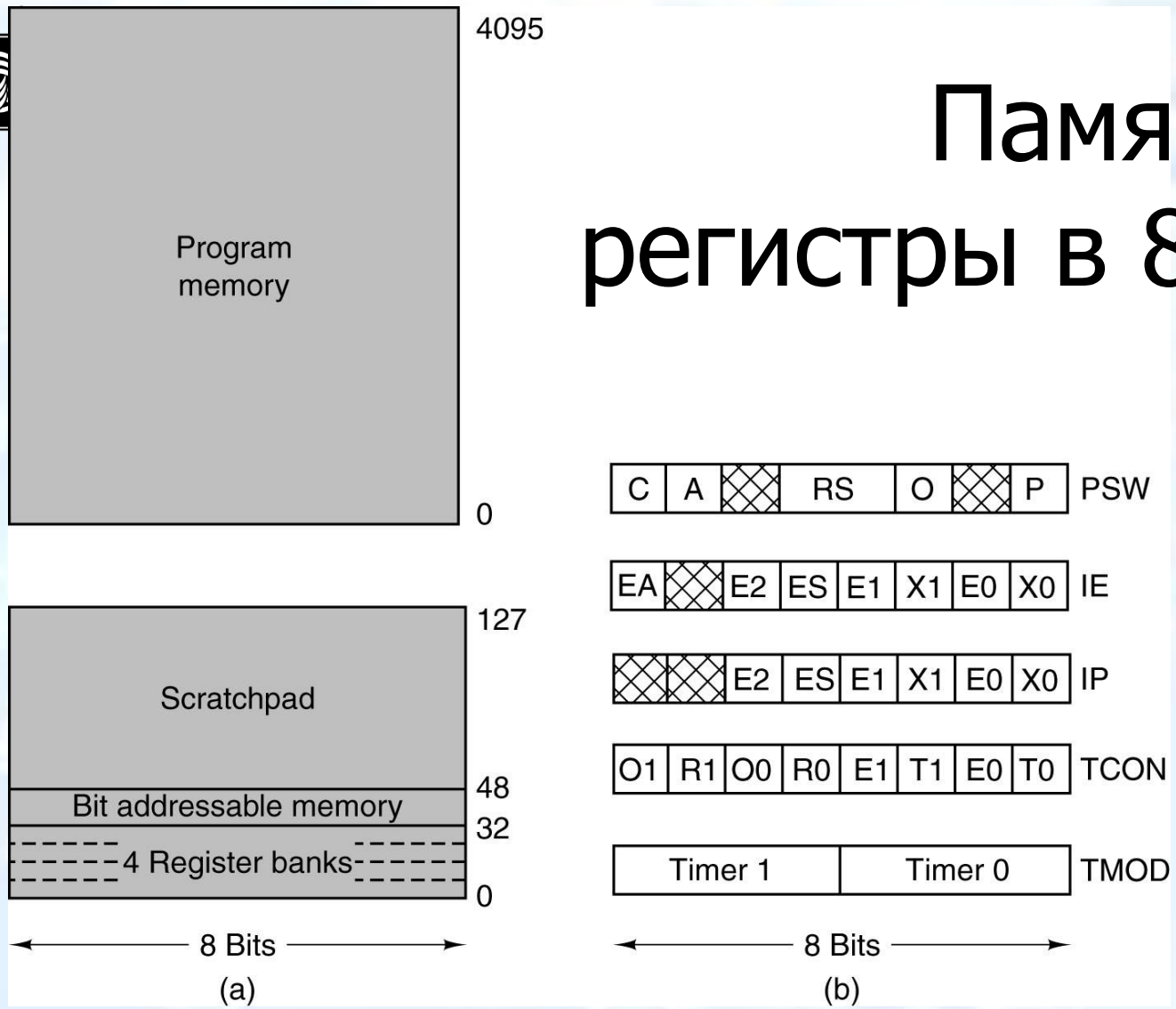




Регистры UltraSPARC III

Register	Alt. name	Function
R0	G0	Hardwired to 0. Stores into it are just ignored.
R1 – R7	G1 – G7	Holds global variables
R8 – R13	O0 – O5	Holds parameters to the procedure being called
R14	SP	Stack pointer
R15	O7	Scratch register
R16 – R23	L0 – L7	Holds local variables for the current procedure
R24 – R29	I0 – I5	Holds incoming parameters
R30	FP	Pointer to the base of the current stack frame
R31	I7	Holds return address for the current procedure

Память и регистры в 8051



Типы данных в Pentium 4

Type	1 Bit	8 Bits	16 Bits	32 Bits	64 Bits	128 Bits
Bit						
Signed integer		×	×	×		
Unsigned integer		×	×	×		
Binary coded decimal integer		×				
Floating point				×	×	

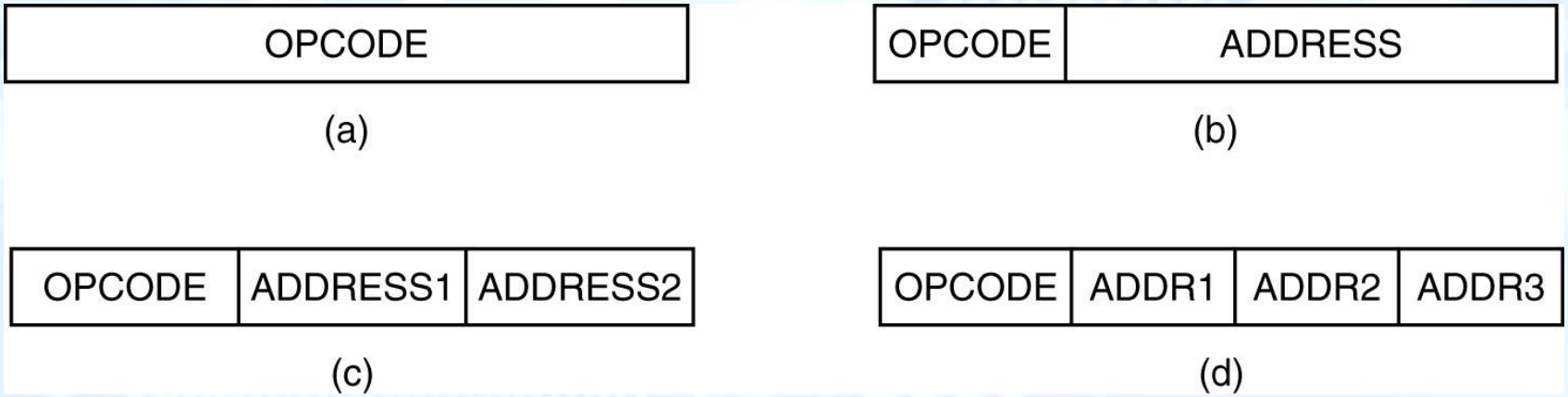
Типы данных в UltraSPARC III

Type	1 Bit	8 Bits	16 Bits	32 Bits	64 Bits	128 Bits
Bit						
Signed integer		×	×	×	×	
Unsigned integer		×	×	×	×	
Binary coded decimal integer						
Floating point				×	×	×

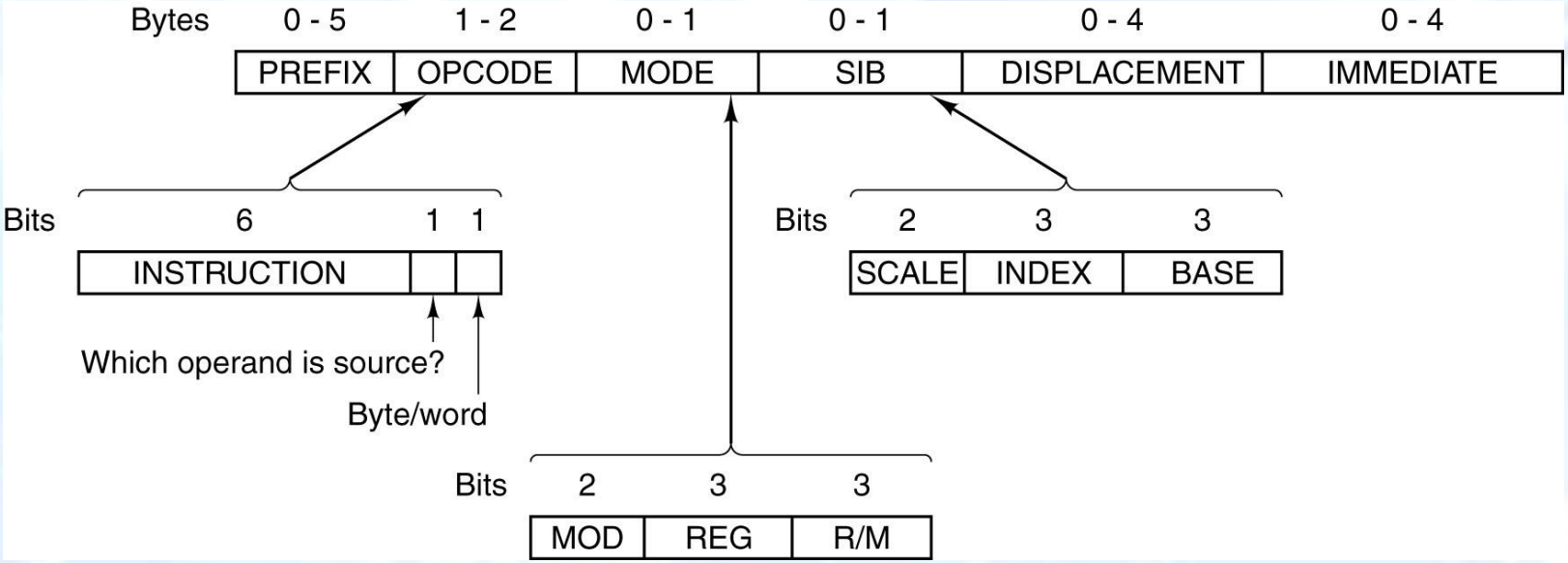
Типы данных в 8051

Type	1 Bit	8 Bits	16 Bits	32 Bits	64 Bits	128 Bits
Bit	×					
Signed integer		×				
Unsigned integer						
Binary coded decimal integer						
Floating point						

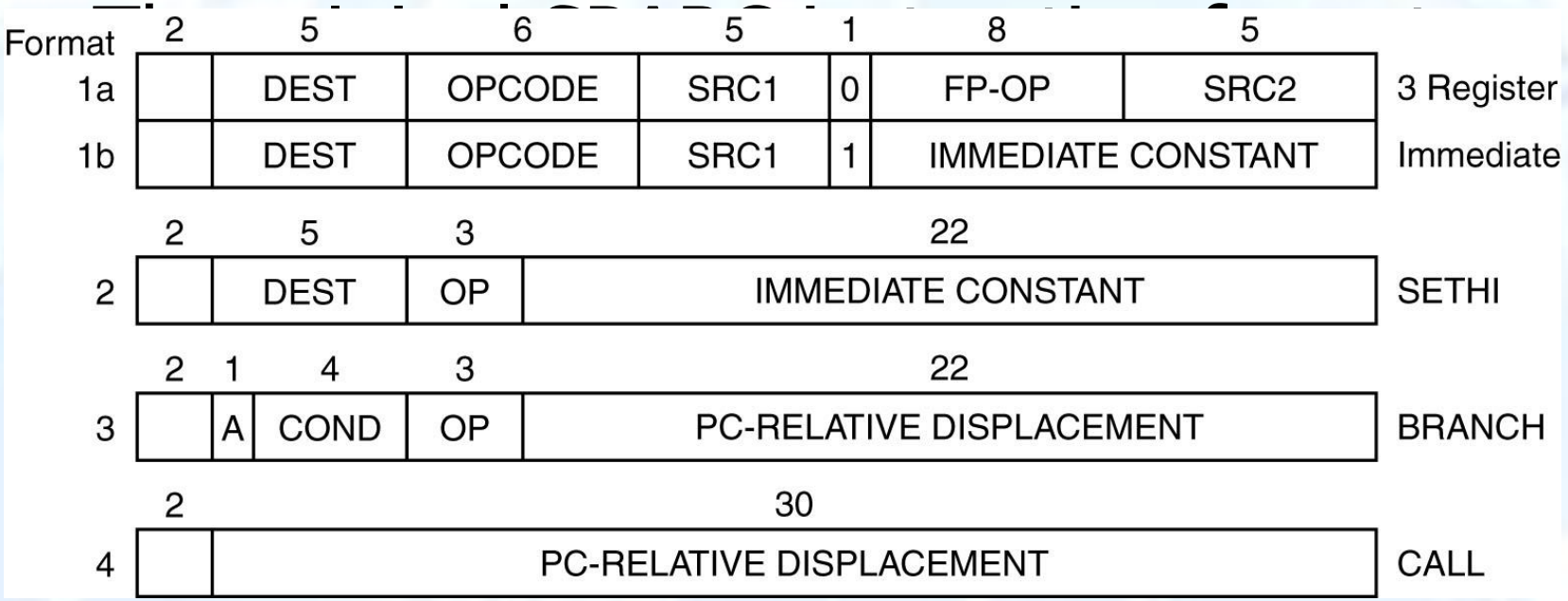
Возможные форматы команд



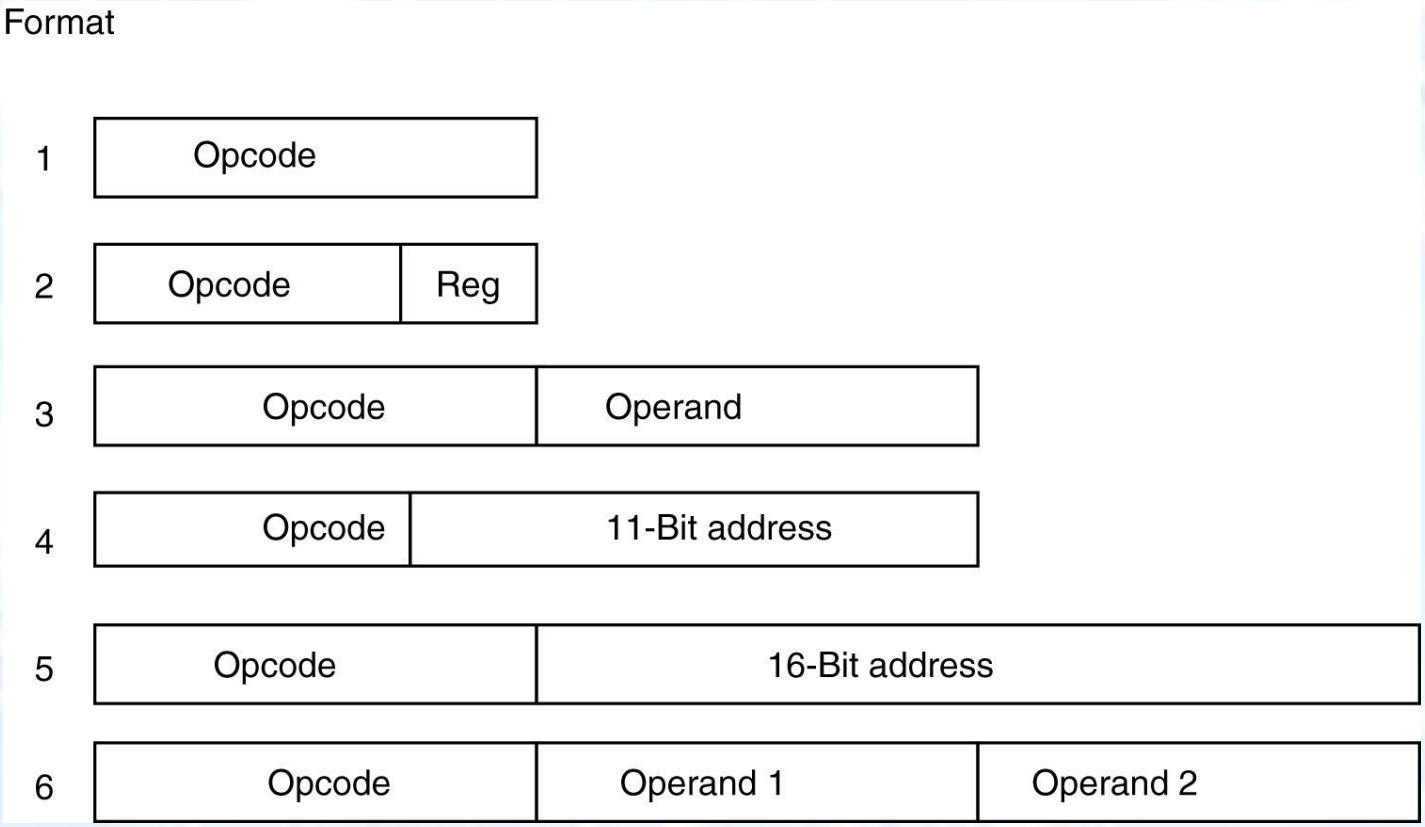
Формат команд Pentium 4

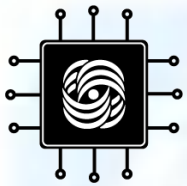


Формат команд UltraSPARC III



Форматы команд в 8051





Адресация

- Неявная (IADD)
- Непосредственная (MOV R1, 4)
- Прямая (MOV R1, FULL_ADDR)
- Регистровая (MOV R1, R2)
- Косвенная регистровая (MOV R1, [R2])
- Индексная (MOV R1, A[R2])
- Относительная индексная (MOV R1, A[R2+R3])
- Стековая (SADD)

Виды адресации в рассматриваемых архитектурах

Addressing mode	Pentium 4	UltraSPARC III	8051
Accumulator			×
Immediate	×	×	×
Direct	×		×
Register	×	×	×
Register indirect	×	×	×
Indexed	×	×	
Based-indexed		×	
Stack			

Управление циклами

For (I = 0; i<n; i++) {операторы}

```
        i = 1;
L1:     first-statement;
        .
        .
        .
        last-statement;
        i = i + 1;
        if (i < n) goto L1;
```

(a)

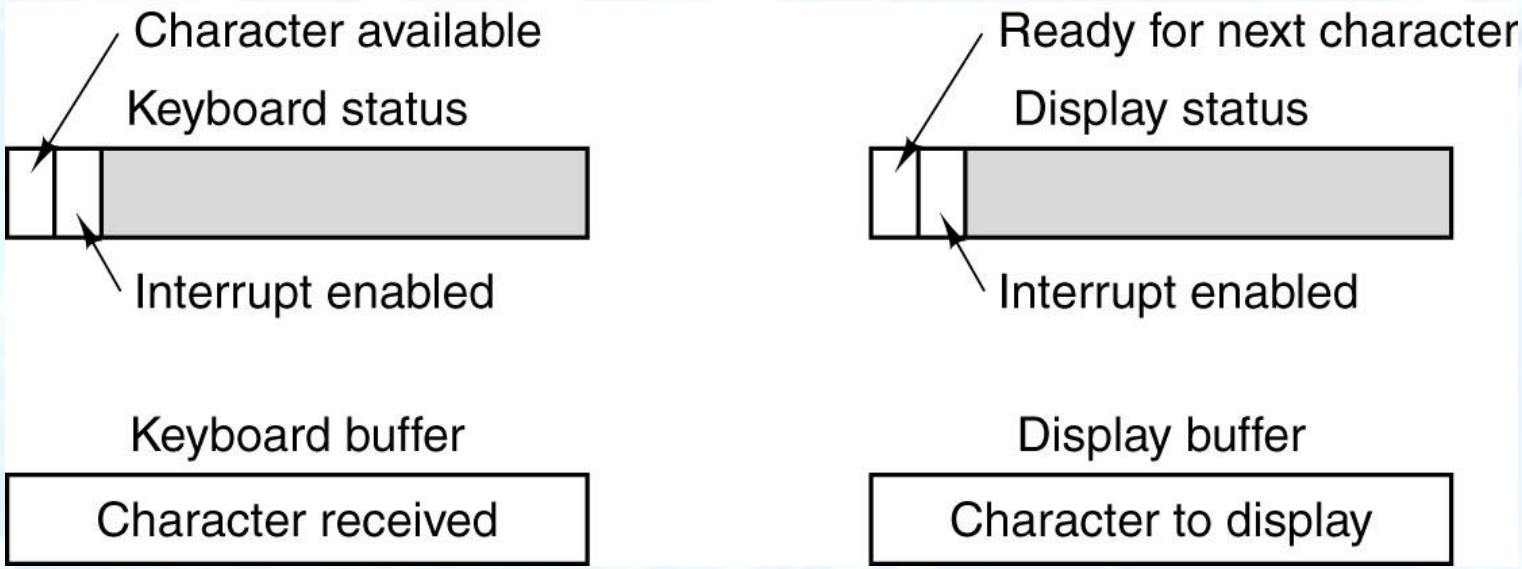
```
        i = 1;
L1:     if (i > n) goto L2;
        first-statement;
        .
        .
        .
        last-statement
        i = i + 1;
        goto L1;
L2:
```

(b)

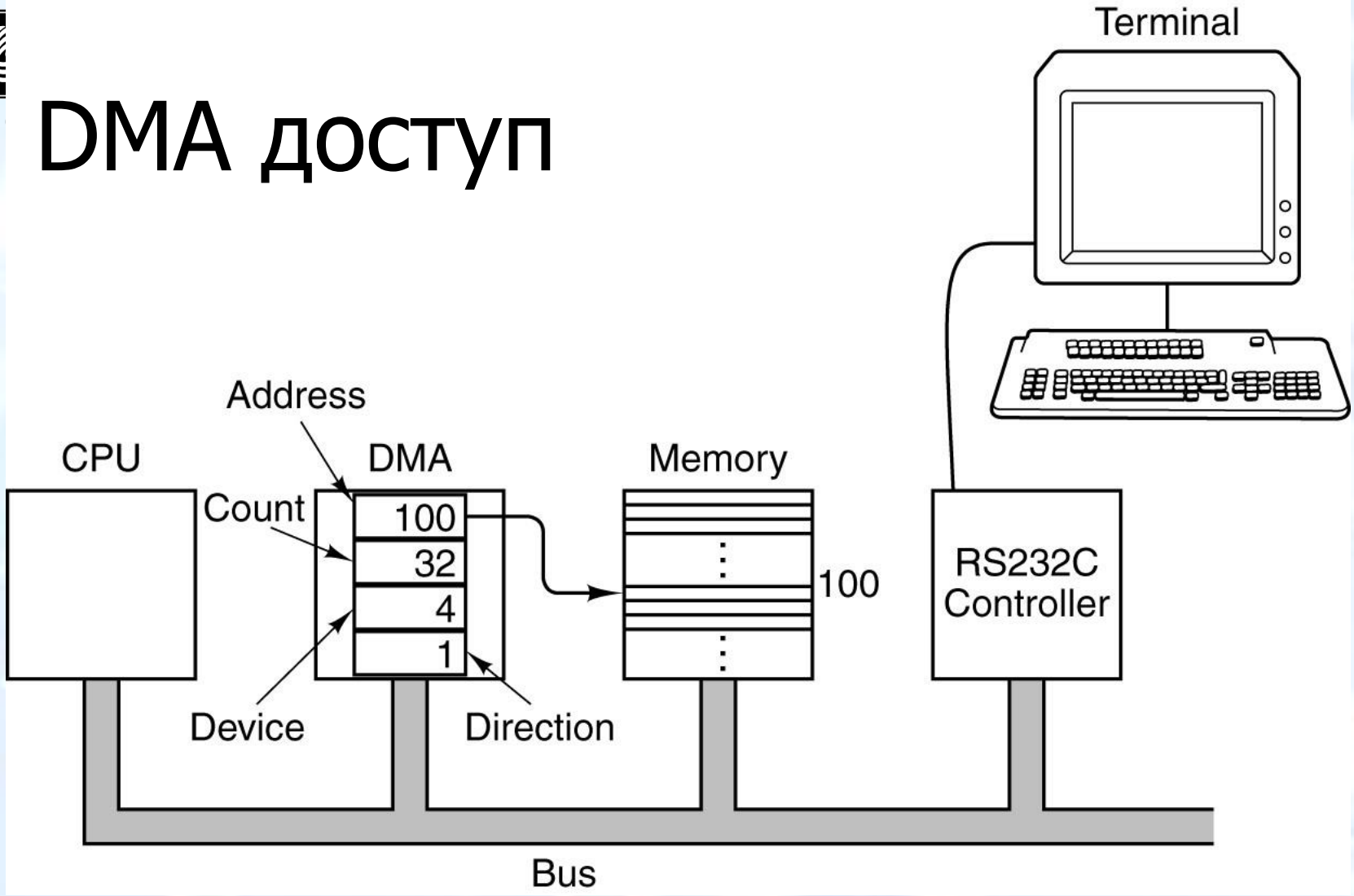
Команды ввода-вывода

- Программируемый ввод-вывод с активным ожиданием
- Ввод-вывод с управлением по прерываниям
- Ввод-вывод с прямым доступом к памяти

Регистры устройств



DMA доступ


Terminal
Address
CPU
DMA
Memory
Count
100
32
4
1
100
RS232C
Controller
Device
Direction
Bus

Инструкции The Pentium 4 (1)

Moves

MOV DST,SRC	Move SRC to DST
PUSH SRC	Push SRC onto the stack
POP DST	Pop a word from the stack to DST
XCHG DS1,DS2	Exchange DS1 and DS2
LEA DST,SRC	Load effective addr of SRC into DST
CMOVcc DST,SRC	Conditional move

Arithmetic

ADD DST,SRC	Add SRC to DST
SUB DST,SRC	Subtract SRC from DST
MUL SRC	Multiply EAX by SRC (unsigned)
IMUL SRC	Multiply EAX by SRC (signed)
DIV SRC	Divide EDX:EAX by SRC (unsigned)
IDIV SRC	Divide EDX:EAX by SRC (signed)
ADC DST,SRC	Add SRC to DST, then add carry bit
SBB DST,SRC	Subtract SRC & carry from DST
INC DST	Add 1 to DST
DEC DST	Subtract 1 from DST
NEG DST	Negate DST (subtract it from 0)

Инструкции The Pentium 4 (2)

Binary coded decimal

DAA	Decimal adjust
DAS	Decimal adjust for subtraction
AAA	ASCII adjust for addition
AAS	ASCII adjust for subtraction
AAM	ASCII adjust for multiplication
AAD	ASCII adjust for division

Boolean

AND DST,SRC	Boolean AND SRC into DST
OR DST,SRC	Boolean OR SRC into DST
XOR DST,SRC	Boolean Exclusive OR SRC to DST
NOT DST	Replace DST with 1's complement

Shift/rotate

SAL/SAR DST,#	Shift DST left/right # bits
SHL/SHR DST,#	Logical shift DST left/right # bits
ROL/ROR DST,#	Rotate DST left/right # bits
RCL/RCR DST,#	Rotate DST through carry # bits

Инструкции The Pentium 4 (3)

Test/compare

TEST SRC1,SRC2	Boolean AND operands, set flags
CMP SRC1,SRC2	Set flags based on SRC1 - SRC2

Transfer of control

JMP ADDR	Jump to ADDR
Jxx ADDR	Conditional jumps based on flags
CALL ADDR	Call procedure at ADDR
RET	Return from procedure
IRET	Return from interrupt
LOOPxx	Loop until condition met
INT n	Initiate a software interrupt
INTO	Interrupt if overflow bit is set

Strings

LODS	Load string
STOS	Store string
MOVS	Move string
CMPS	Compare two strings
SCAS	Scan Strings

Инструкции The Pentium 4 (4)

Condition codes

STC	Set carry bit in EFLAGS register
CLC	Clear carry bit in EFLAGS register
CMC	Complement carry bit in EFLAGS
STD	Set direction bit in EFLAGS register
CLD	Clear direction bit in EFLAGS reg
STI	Set interrupt bit in EFLAGS register
CLI	Clear interrupt bit in EFLAGS reg
PUSHFD	Push EFLAGS register onto stack
POPFD	Pop EFLAGS register from stack
LAHF	Load AH from EFLAGS register
SAHF	Store AH in EFLAGS register

Miscellaneous

SWAP DST	Change endianness of DST
CWQ	Extend EAX to EDX:EAX for division
CWDE	Extend 16-bit number in AX to EAX
ENTER SIZE,LV	Create stack frame with SIZE bytes
LEAVE	Undo stack frame built by ENTER
NOP	No operation
HLT	Halt
IN AL,PORT	Input a byte from PORT to AL
OUT PORT,AL	Output a byte from AL to PORT
WAIT	Wait for an interrupt

SRC = source
DST = destination
# = shift/rotate count
LV = # locals

The UltraSPARC III Инструкции (1)

Loads

LDSB ADDR,DST	Load signed byte (8 bits)
LDUB ADDR,DST	Load unsigned byte (8 bits)
LDSH ADDR,DST	Load signed halfword (16 bits)
LDUH ADDR,DST	Load unsigned halfword (16)
LDSW ADDR,DST	Load signed word (32 bits)
LDUW ADDR,DST	Load unsigned word (32 bits)
LDX ADDR,DST	Load extended (64-bits)

Stores

STB SRC,ADDR	Store byte (8 bits)
STH SRC,ADDR	Store halfword (16 bits)
STW SRC,ADDR	Store word (32 bits)
STX SRC,ADDR	Store extended (64 bits)

The UltraSPARC III Инструкции (2)

Arithmetic	
ADD R1,S2,DST	Add
ADDCC "	Add and set icc
ADDC "	Add with carry
ADDCCC "	Add with carry and set icc
SUB R1,S2,DST	Subtract
SUBCC "	Subtract and set icc
SUBC "	Subtract with carry
SUBCCC "	Subtract with carry and set icc
MULX R1,S2,DST	Multiply
SDIVX R1,S2,DST	Signed divide
UDIVX R1,S2,DST	Unsigned divide
TADCC R1,S2,DST	Tagged add

The UltraSPARC III Инструкции (3)

Shifts/rotates

SLL R1,S2,DST	Shift left logical (32 bits)
SLLX R1,S2,DST	Shift left logical extended (64)
SRL R1,S2,DST	Shift right logical (32 bits)
SRLX R1,S2,DST	Shift right logical extended (64)
SRA R1,S2,DST	Shift right arithmetic (32 bits)
SRAX R1,S2,DST	Shift right arithmetic ext. (64)

Miscellaneous

SETHI CON,DST	Set bits 10 to 31
MOVcc CC,S2,DST	Move on condition
MOVr R1,S2,DST	Move on register
NOP	No operation
POPC S1,DST	Population count
RDCCR V,DST	Read condition code register
WRCCR R1,S2,V	Write condition code register
RDPC V,DST	Read program counter

The UltraSPARC III Инструкции (4)

Boolean	
AND R1,S2,DST	Boolean AND
ANDCC "	Boolean AND and set icc
ANDN "	Boolean NAND
ANDNCC "	Boolean NAND and set icc
OR R1,S2,DST	Boolean OR
ORCC "	Boolean OR and set icc
ORN "	Boolean NOR
ORNCC "	Boolean NOR and set icc
XOR R1,S2,DST	Boolean XOR
XORCC "	Boolean XOR and set icc
XNOR "	Boolean EXCLUSIVE NOR
XNORCC "	Boolean EXCL. NOR and set icc

The UltraSPARC III Инструкции (5)

Transfer of control	
BPcc ADDR	Branch with prediction
BPr SRC,ADDR	Branch on register
CALL ADDR	Call procedure
RETURN ADDR	Return from procedure
JMPL ADDR,DST	Jump and link
SAVE R1,S2,DST	Advance register windows
RESTORE "	Restore register windows
Tcc CC,TRAP#	Trap on condition
PREFETCH FCN	Prefetch data from memory
LDSTUB ADDR,R	Atomic load/store
MEMBAR MASK	Memory barrier

SRC = source register
DST = destination register
R1 = source register
S2 = source: register or immediate
ADDR = memory address

TRAP# = trap number
FCN = function code
MASK = operation type
CON = constant
V = register designator

CC = condition code set
R =destination register
cc = condition
r = LZ,LEZ,Z,NZ,GZ,GEZ

The UltraSPARC III Инструкции (6)

Instruction	How to do it
MOV SRC,DST	OR SRC with G0 and store the result DST
CMP SRC1,SRC2	SUBCC SRC2 from SRC1 and store the result in G0
TST SRC	ORCC SRC with G0 and store the result in G0
NOT DST	XNOR DST with G0
NEG DST	SUB DST from G0 and store in DST
INC DST	ADD 1 to DST (immediate operand)
DEC DST	SUB 1 from DST (immediate operand)
CLR DST	OR G0 with G0 and store in DST
NOP	SETHI G0 to 0
RET	JMPL %I7+8,%G0

8051 Инструкции (1)

Inst.	Description	ACC	Reg	Dir	@R	#	C	Bit
MOV	Move *src* to ACC		×	×	×	×		
MOV	Move *src* to register	×		×		×		
MOV	Move *src* to memory	×	×	×	×	×		
MOV	Move *src* to indirect RAM	×		×		×		
MOV	Move 16-bit constant to DPTR							
MOVC	Move code to ACC offset from DPTR							
MOVC	Move code to ACC offset from PC							
MOVX	Move external RAM byte to ACC				×			
MOVX	Move ext. RAM byte to ACC @DPTR							
MOVX	Move to ext. RAM byte from ACC				×			
MOVX	Move to ext. RAM byte from ACC @DPTR							
PUSH	Push *src* byte to stack			×				
POP	Pop stack byte to dst			×				
XCH	Exchange ACC and dst	×		×	×			
XCHD	Exchange low-order digit ACC and dst			×				
SWAP	Swap nibbles of dst	×						
ADD	Add *src* to ACC		×	×	×	×		

8051 Инструкции (2)

Inst.	Description	ACC	Reg	Dir	@R	#	C	Bit
ADDC	Add *src* to ACC with carry		×	×	×	×		
SUBB	Subtract *src* from ACC with borrow		×	×	×	×		
INC	Increment dst	×	×	×	×			
DEC	Decrement dst	×	×	×	×			
INC	DPTR							
MUL	Multiply							
DIV	Divide							
DA	Decimal adjust dst	×						
ANL	AND *src* to ACC		×	×	×	×		
ANL	AND ACC to dst			×				
ANL	AND immediate to dst			×				
ORL	OR *src* to ACC		×	×	×	×		
ORL	OR ACC to dst			×				
ORL	OR immediate to dst			×				

8051 Инструкции (3)

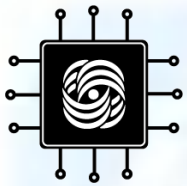
Inst.	Description	ACC	Reg	Dir	@R	#	C	Bit
XRL	XOR *src* to ACC		×	×	×	×		
XRL	XOR ACC to dst			×				
XRL	XOR immediate to dst			×				
CLR	Clear dst	×						
CPL	Complement dst	×						
RL	Rotate dst left	×						
RLC	Rotate dst left through carry	×						
RR	Rotate dst right	×						
RRC	Rotate dst right through carry	×						

8051 Инструкции (4)

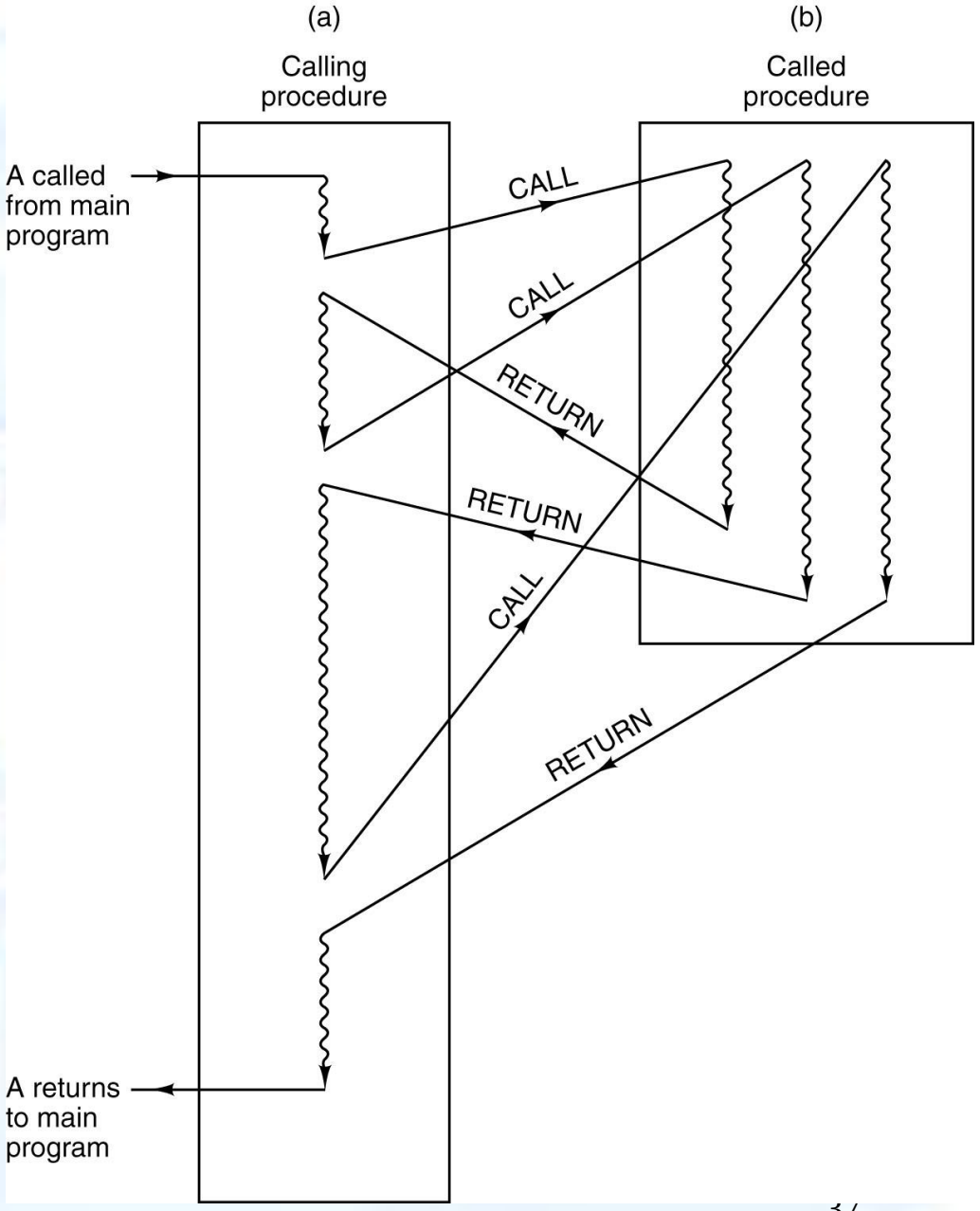
Inst.	Description	ACC	Reg	Dir	@R	#	C	Bit
CLR	Clear bit						×	×
SETB	Set bit						×	×
CPL	Complement bit						×	×
ANL	AND *src* to carry							×
ANL	AND complement of *src* to carry							×
ORL	OR *src* to carry							×
ORL	OR complement of *src* to carry							×
MOV	Move *src* to carry							×
MOV	Move carry to src							×
JV	Jump relative if carry set							
JNC	Jump relative if carry not set							
JB	Jump relative if direct bit set							×
JNB	Jump relative if direct bit not set							×
JBC	Jump rel. if direct bit set and carry clear							×

8051 Инструкции (5)

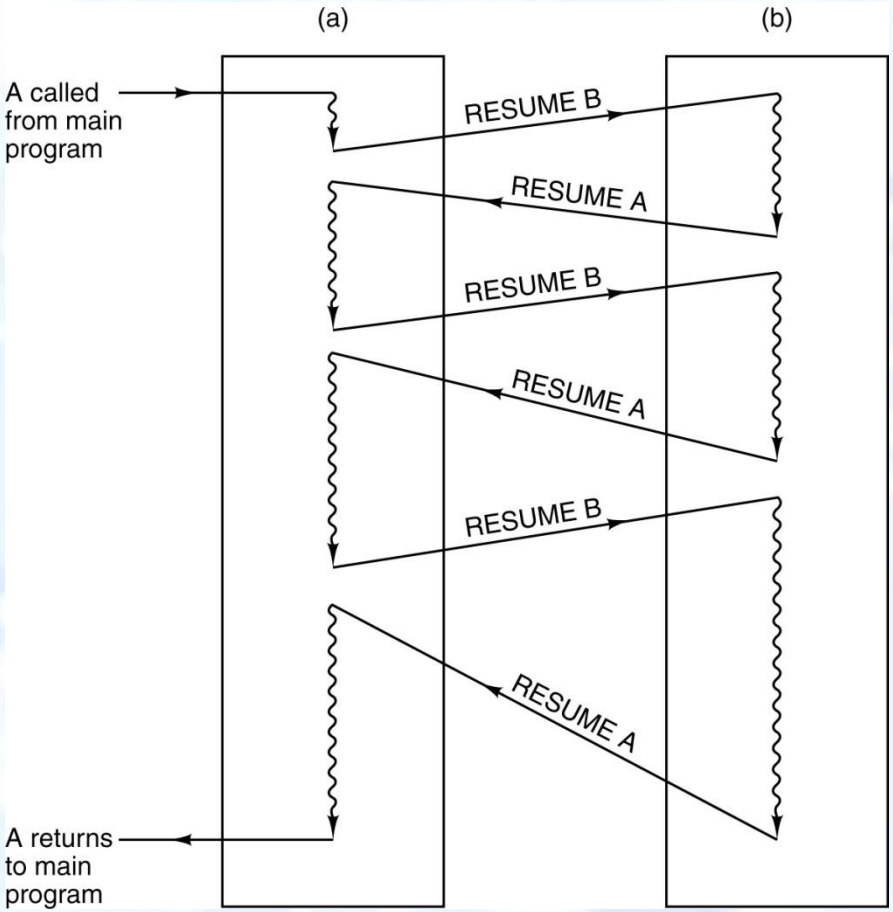
Inst.	Description	ACC	Reg	Dir	@R	#	C	Bit
ACALL	Call subroutine (11-bit addr)							
LCALL	Call subroutine (16-bit addr)							
RET	Return from subroutine							
RETI	Return from interrupt							
SJMP	Short relative jump (8-bit addr)							
AJMP	Absolute jump (11-bit addr)							
LJMP	Absolute jump (16-bit addr)							
JMP	Jump indirect rel. to DPR+ACC							
JZ	Jump if ACC is zero							
JNZ	Jump if ACC is nonzero							
CJNE	Comp. *src* to ACC, jump unequal			×		×		
CJNE	Comp. *src* to immediate, jump unequal		×		×			
DJNZ	Decrement dst and jump nonzero							
NOP	No operation							



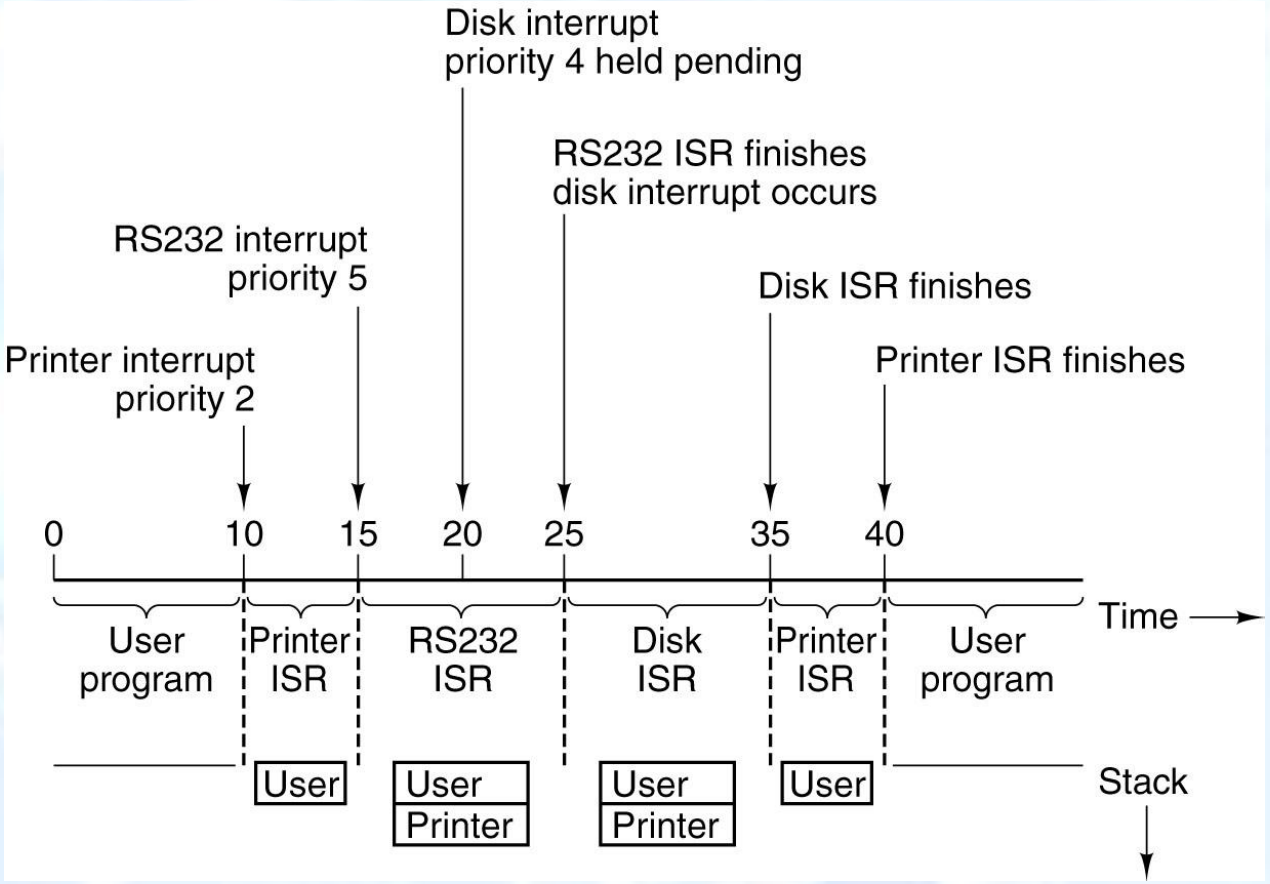
Процедуры (1)


(a)
(b)
Calling procedure
Called procedure
A called from main program
CALL
CALL
RETURN
RETURN
CALL
RETURN
A returns to main program

Сопрограммы (2)



Прерывания


Disk interrupt
priority 4 held pending
RS232 ISR finishes
disk interrupt occurs
RS232 interrupt
priority 5
Disk ISR finishes
Printer interrupt
priority 2
Printer ISR finishes
0
10
15
20
25
35
40
Time
User
program
Printer
ISR
RS232
ISR
Disk
ISR
Printer
ISR
User
program
User
User
Printer
User
Printer
User
Stack

Проблемы Pentium 4

- CISC-архитектура
- Ориентирована на 2-х адресные команды
- Мало регистров
- Команды не выполняются подряд
- Неточное предсказание переходов
- Спекулятивное выполнение
- Ограничение размера программ в 4Гб

Спасибо за внимание!