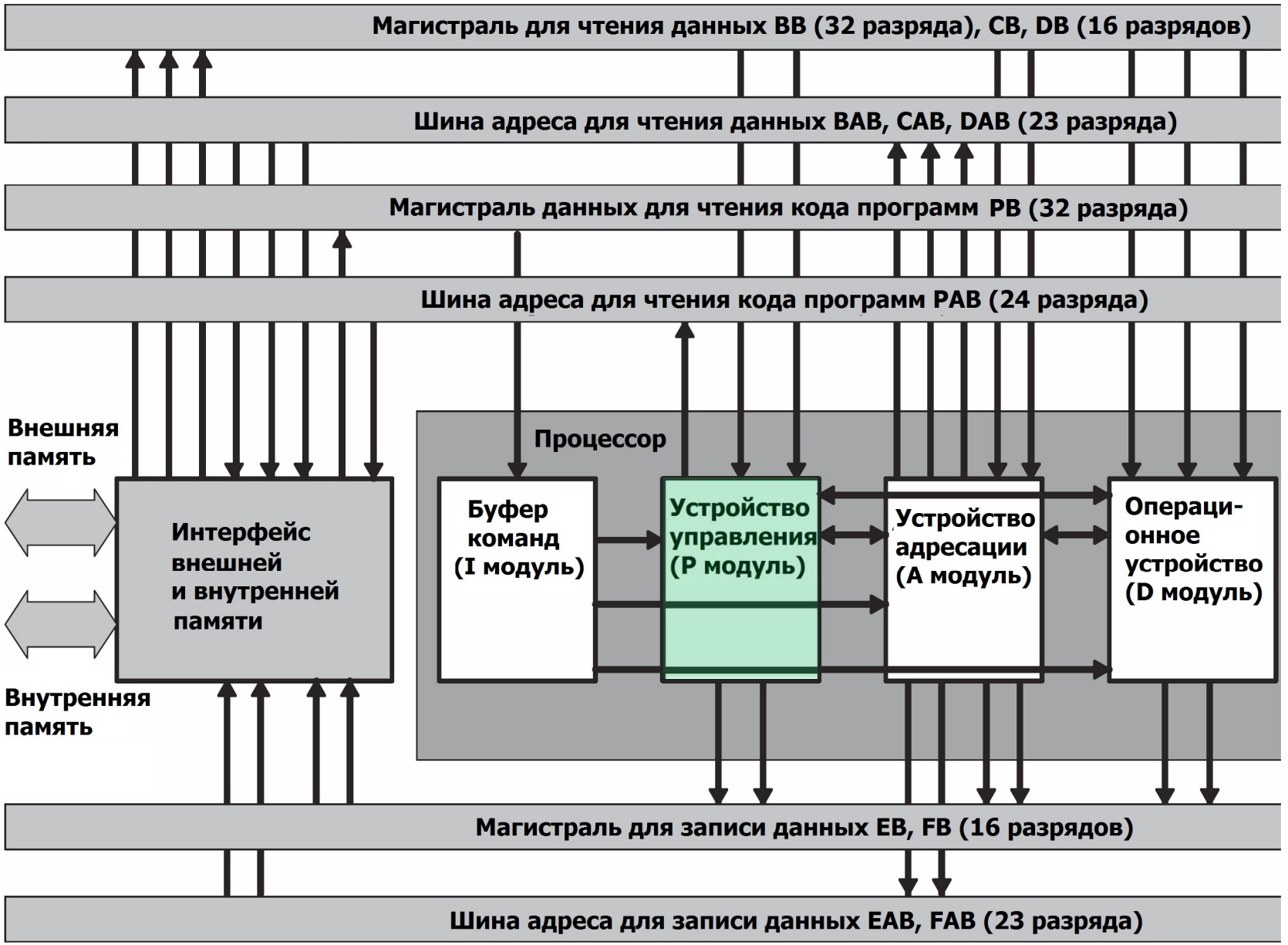
Микропроцессорные устройства обработки сигналов

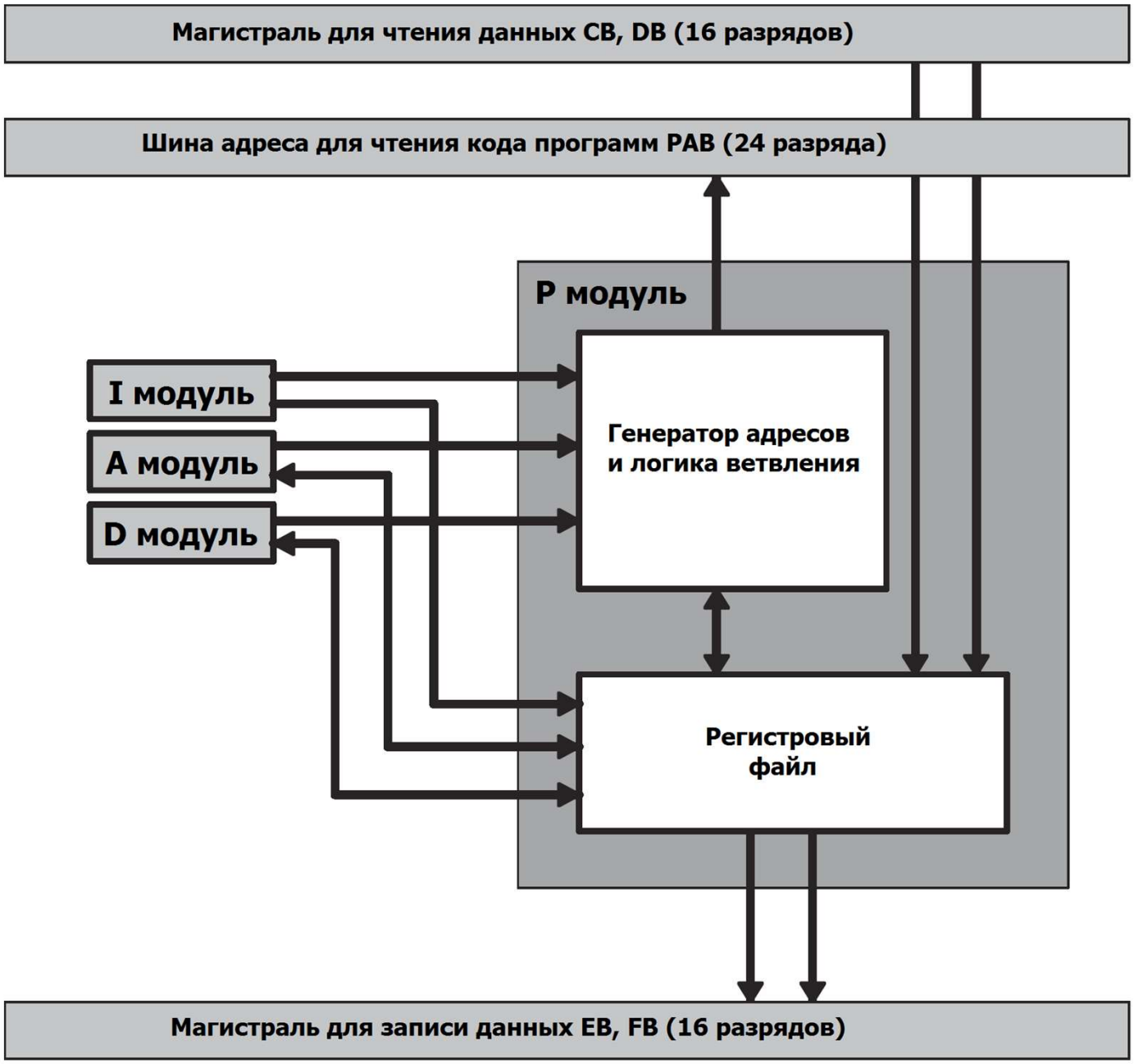
Лекция L08
«Устройство управления»

http://vykhovanets.ru/course67/

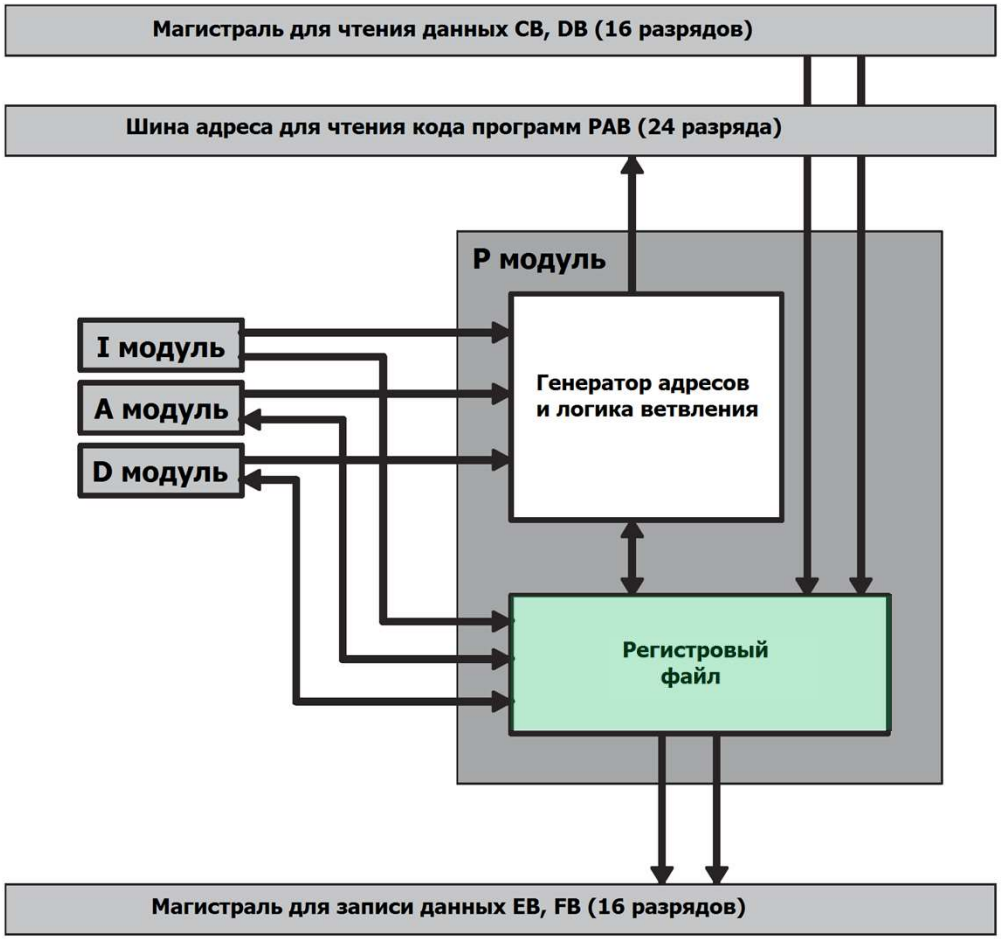
Ядро микропроцессора



Устройство управления



Регистровый файл P

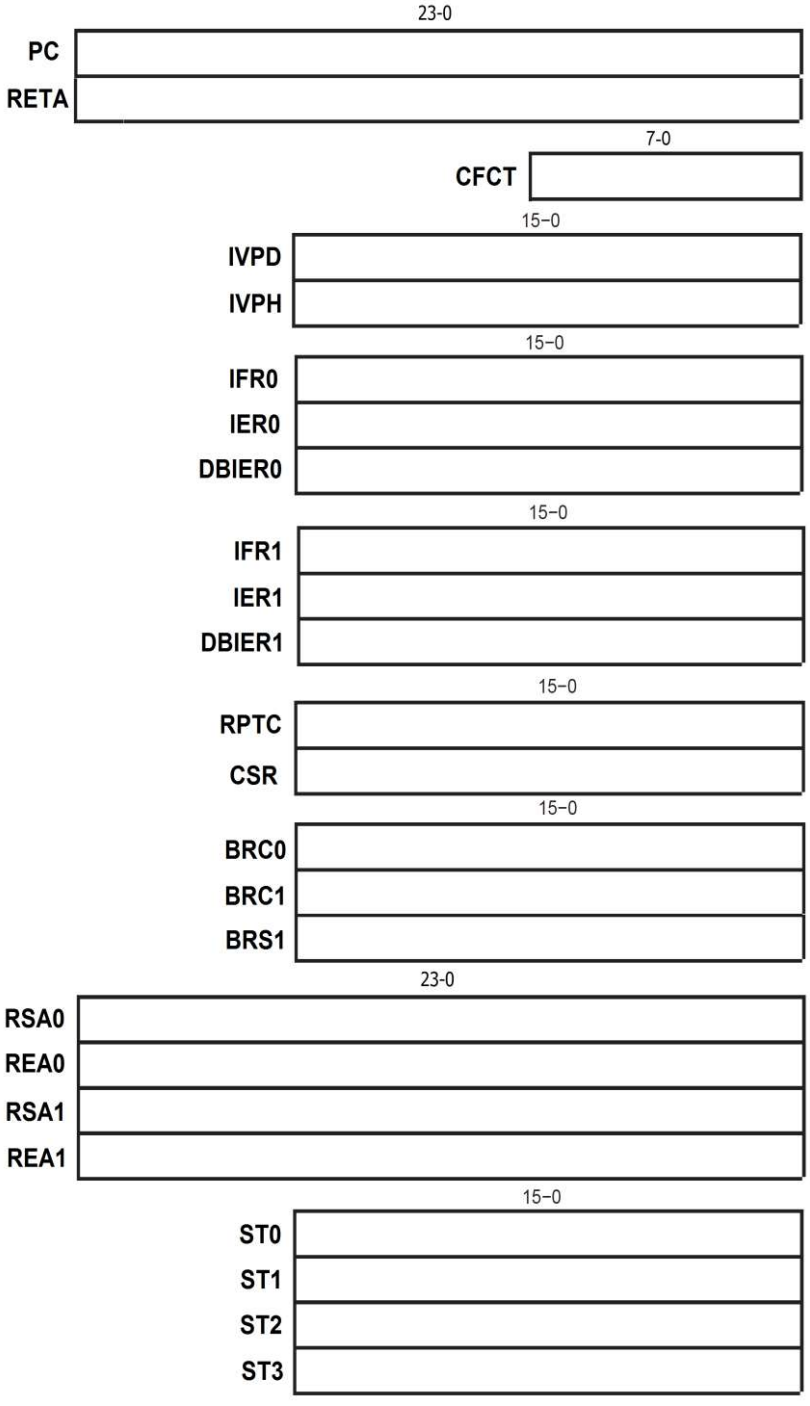


- Хранение промежуточных данных P модуля
- Двухсторонний обмен данными с генератором
- Обмен промежуточными данными с модулями A, D и памятью

- **Регистры потока команд**: **PC**; **RETA**; **CFCT**.
- **Регистры статуса**: **ST0 - ST3**.
- **Регистры повторения команды**: **RPTC**; **CSR**.

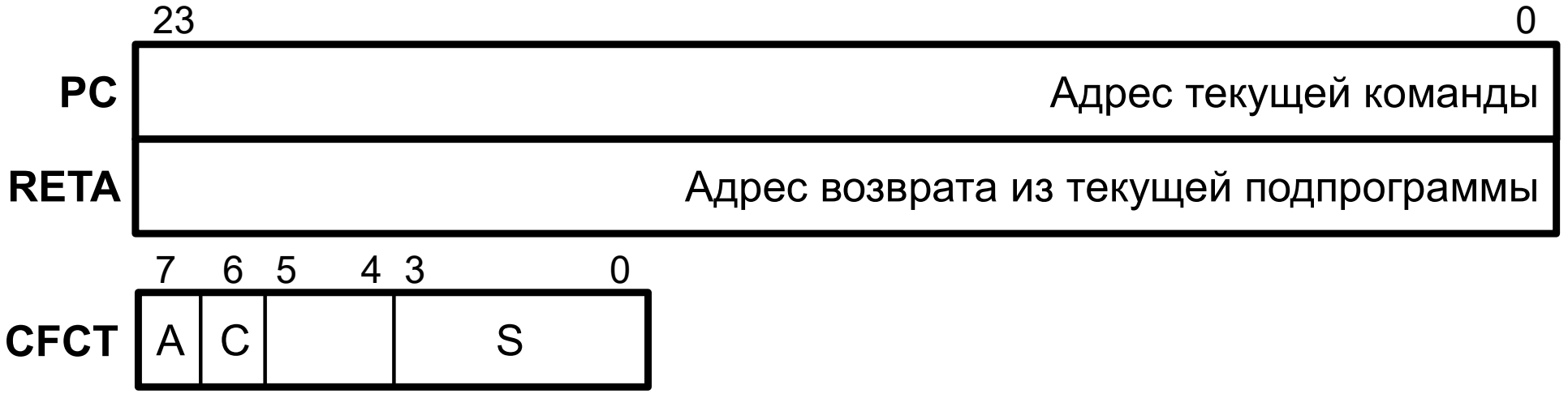
- **Регистры повторения блока**: **BRC0, BRC1**; **BRS1**; **RSA0, RSA1**; **REA0, REA1**.
- **Регистры прерываний**: **IVPD**, **IVPH**; **IFR0, IFR1**; **IER0, IER1**; **DBIER0, DBIER1**.

Регистры P



- **Регистры потока команд**:
  PC – **P**rogram **C**ounter (счетчик команд);
  RETA – **RET**urn **A**ddress (адреса возврата);
  CFCT – **C**ontrol **F**low **C**on**T**ext (контекст потока управления).
- **Регистры статуса**:
  ST0 - ST3 – **St**atus (состояние микропроцессора).
- **Регистры повторения команды**:
  RPTC – **R**e**P**ea**T** **C**ounter (счетчик повторения);
  CSR – **C**omputed **S**ingle-**R**epeat (число повторений).
- **Регистры повторения блока**:
  BRC0, BRC1 – **B**lock-**R**epeat **C**ounter (счетчик повторения блока команд);
  BRS1 – **B**lock-**R**epeat **S**ave (сохранение BRC1);
  RSA0, RSA1 – **R**epeat **S**tart **A**ddress (начальный адрес блока команд);
  REA0, REA1 – **R**epeat **E**nd **A**ddress (конечный адрес блока команд).
- **Регистры прерываний**:
  IVPD, **IVPH** – **I**nterrupt **V**ector **P**age **D**, **H** (страницы векторов прерываний);
  IFR0, IFR1 – **I**nterrupt **F**lag **R**egister (флаги прерываний);
  IER0, IER1 – **I**nterrupt **E**nable **R**egister (разрешения прерываний);
  DBIER0, DBIER1 – **D**ebug **I**nterrupt **E**nable **R**egister.

Регистры потока команд



- **CFCT** – Control Flow Context (контекст потока управления):
  – **A** – Active (активность повторения текущей команды);
  – **C** – Condition (состояние условия повторения текущей команды);
  – **S** – Status (состояния двух уровней блочного повторения команд): активности, локальность (блок помещен в очередь команд или нет).

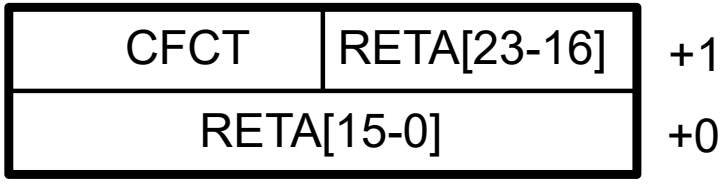
MOV RETA, dbl(Lmem)

```
1110 1011 AAAA AAAI xxxx 01xx
```

MOV dbl(Lmem), **RETA**

```
1110 1101 AAAA AAAI xxxx 011x
```

dbl(Lmem)



Регистры простого повторения

MOV RPTC, TAx

```
0100010E 1110FDDD
```

MOV Smem, **CSR**

```
1101 1100 AAAA AAAI x000 xx11
```

MOV CSR, Smem

```
1110 0101 AAAA AAAI x000 11xx
```

MOV k12, **CSR**

```
0001 011E kkkk kkkk kkkk 1000
```

MOV TAx, **CSR**

```
0101 001E FSSS 1100
```

RPT CSR

```
0100 100E | xxxx x000
```

RPT CSR

MAC *AR3+, *AR4+, AC1

$$AC1 = \sum_{k=0}^{CSR} AR3_k * AR4_k$$

E – Parallel **E**nable Bit (бит разрешения параллельного выполнения команды).
MAC – Multiply and Accumulate (умножение с накоплением).

Регистры блочного повторения

	15 … 0	
BRC0	00 001Ah	Счетчик повторения блока 0
BRC1	00 0039h	Счетчик повторений блока 1
BRS1	00 003Ah	Регистр для хранения BRC1

	23 … 0	
RSA0	00 001Bh	Начальный адрес блока 0
REA0	00 001Ch	Конечный адрес блока 0
RSA1	00 003Ch	Начальный адрес блока 1
REA1	00 003Dh	Конечный адрес блока 1

RPTB pmad

```
0000  111E | 1111  1111 | 1111  1111
```

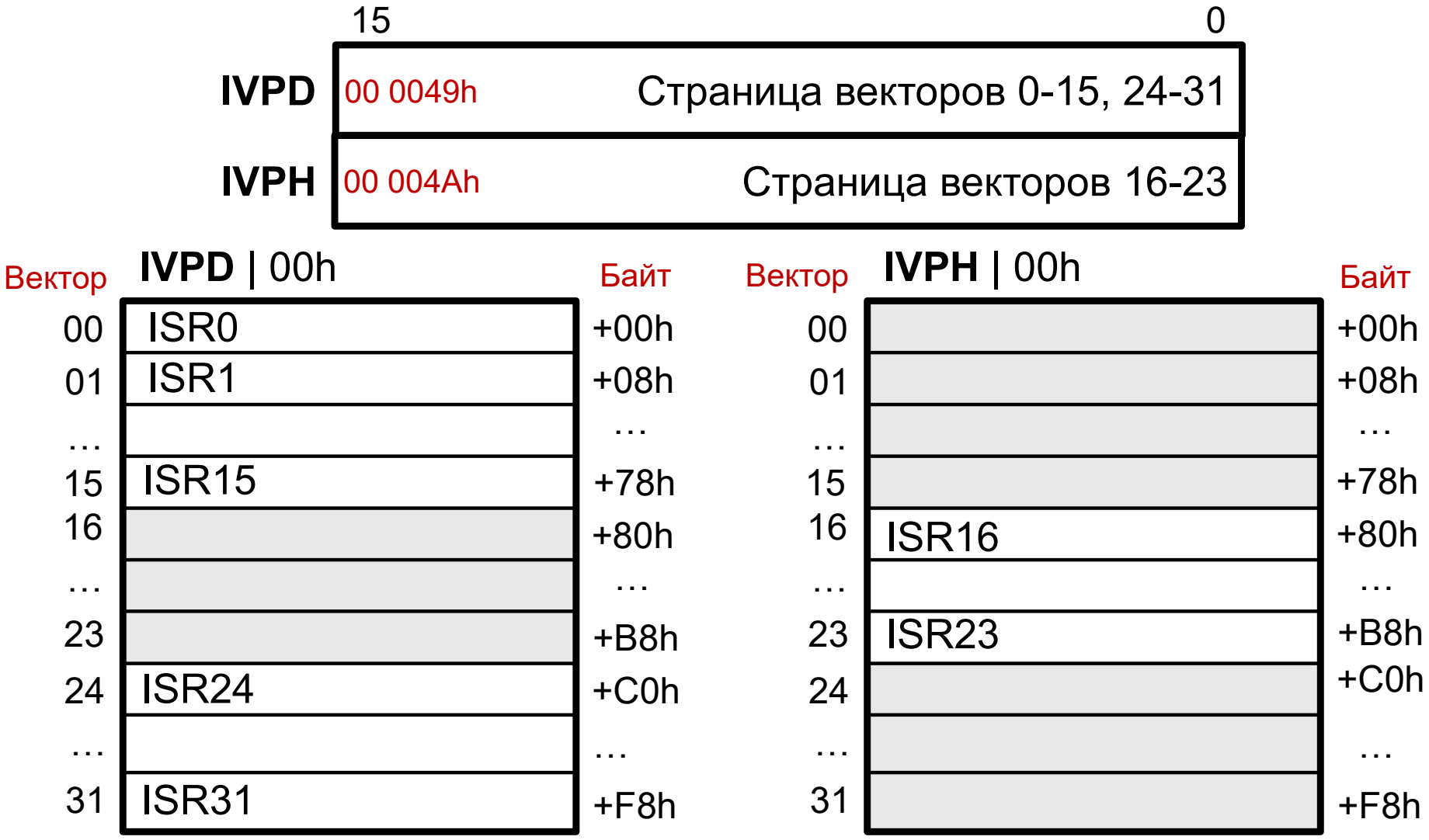
RPTBLOCAL pmad

```
0100  101E | 1111  1111
```

pmad – **p**rogra**m** **ad**dress (прогр. адрес).
l – **l**abel (беззнаковое смещение относительно PC).

```
MOV #3, BRC0 ; повтор 4 раза
MOV #1, BRC1 ; повтор 2 раза, BRC1→ BRS1
RPT { ; блок 0
   RPTLOCAL { ; BRS1→ BRC1, блок 1
    … ; команды
   } ; если BRC1 не ноль – в начало и BRC1--
… ; команды
} ; если BRC0 не ноль – в начало и BRC0--
```

Регистры страниц прерываний



ISRx – **I**nterrupt **S**ervice **R**outine x (двойное слово с адресом процедуры обработки прерывания и конфигурацией стека).

Регистры прерываний

IER0 (IFR0, DBIER0)

15	14	13	12	11	10	9	8
INT15	INT14	INT13	INT12	INT11	INT10	INT9	INT8
R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0

7	6	5	4	3	2	1	0
INT7	INT6	INT5	INT4	INT3	INT2	Резерв	
R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R–0	

IER1 (IFR1, DBIER1)

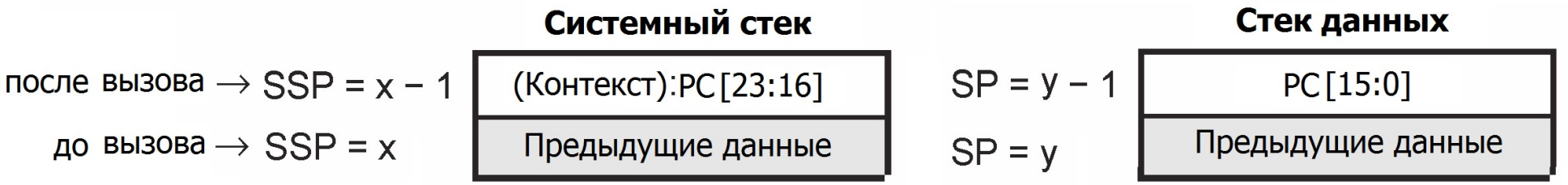
15	11	10	9	8
Резерв		RTOSINT	DLOGINT	BERRINT
R–0		R/W–0	R/W–0	R/W–0

7	6	5	4	3	2	1	0
INT23	INT22	INT21	INT20	INT19	INT18	INT17	INT16
R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0

INT2-INT23 – **I**nterrupt **2-23** (флаги прерываний 2-23).
RTOSINTF – **R**eal **T**ime **O**peration **S**ystem **Int**errupt **F**lag (флаг прерывания операционной системы реального времени).
DLOGINTF – **D**ata log **Int**errupt **F**lag (флаг прерывания протоколирования данных).
BERRINTF – **B**us **Err**or **Int**errupt **F**lag (флаги прерывания по ошибке обмена по шинам и магистралям).
DBIERx – **D**ebug **I**nterrupt **E**nable **R**egister (регистры аппаратурной отладки, флаги указывают, какие прерывания вызывают переход в отладочный режим)

Конфигурация стеков

Медленный вызов и возврат NO_RETA (ISR DAXX:XXXXh)



а) вызов процедуры и возврат из процедуры

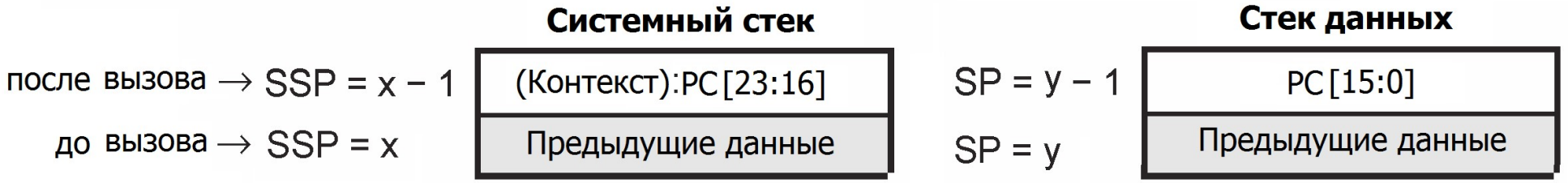
б) вызов прерывания и возврат из прерывания

Быстрый вызов и возврат USE_RETA (ISR CAXX:XXXXh)

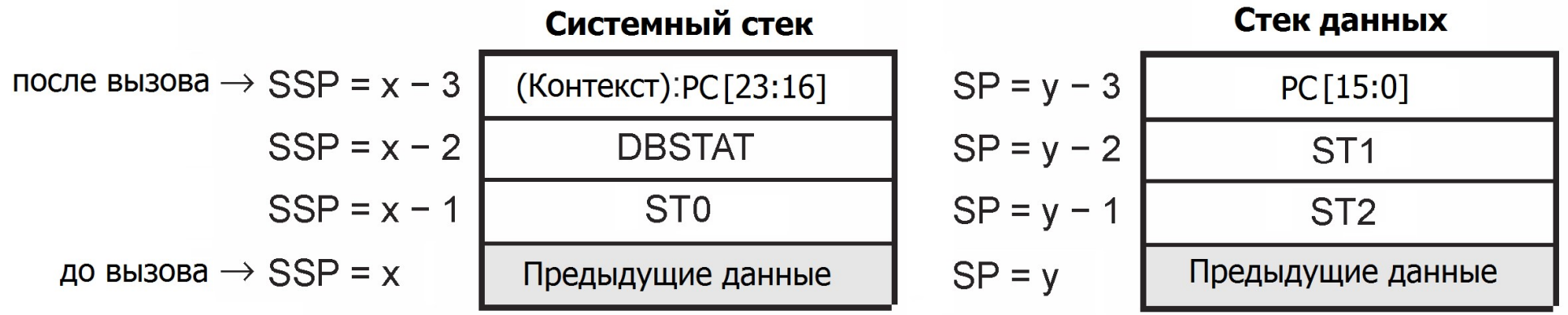
CALL: [PC, Контекст повторения]→[RETA, CFCT] :: [RETA, CFCT]→[*-SP, *-SSP]

RET: [RETA, CFCT]→[PC, Контекст повторения] :: [*SP-, *SSP-]→[RETA, CFCT]

Вызов и возврат

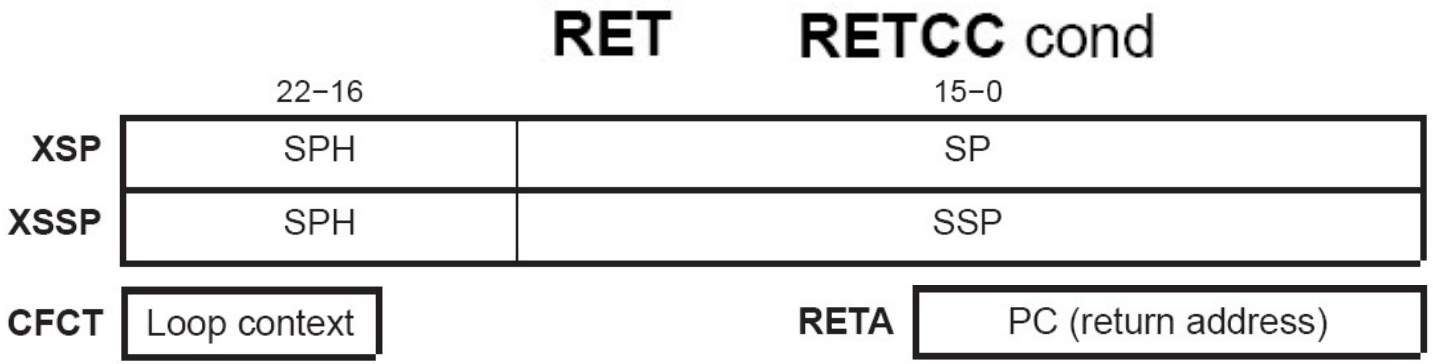


а) вызов процедуры и возврат из процедуры



б) вызов прерывания и возврат из прерывания

CALL ACx
CALL L16
CALL P24
CALLCC L16, cond
CALLCC P24, cond



Соглашения о вызове

- **cdecl** - аргументы передаются через стек ***справа налево***, очистку стека производит ***вызывающая*** функция.
- **stdcall** - аргументы передаются через стек ***справа налево***, очистку стека производит ***вызываемая*** функция.
- **pascal** - аргументы передаются через стек ***слева направо***, очистку стека производит ***вызываемая*** функция.
- **fastcall** - аргументы передаются через регистры и через стек.
- ***r = fun(a1, a2, a3, …);***

Передача аргументов

- Аргументы функции передаются через стек или через регистры в следующем порядке:
  - указатели - (**X**)**AR0,** (**X**)**AR1, …,** (**X**)**AR4**;
  - данные 16 бит - **T0, T1, AR0, …, AR4**;
  - данные 32 или 40 бит - **AC0, AC1, AC2**;
  - аргумент перед многоточием – в стеке.
- Структуры более 32 бита передаются через указатель.
- Аргументы в стеке размещаются в порядке, обратном вызову: *SP(0) – адрес возврата, *SP(1) - 1-й аргумент, …

Возврат результата

- Данные 16 бит – в T0.
- Данные 32 или 40 бит – в AC0.
- Указатель (24) 16 бит – в (X)AR0.
- Структура более 32 бита – в первом аргументе – указатель на структуру, если он 0x0, то структура не возвращается.

struct x func(char, int*, long, long long, int, …)

AR0 – struct x *
T0 – char
AR1 – int*
AC0 – long
AC1 – long long
MOV *SP(#2), T1
MOV *SP(#4), AC2

В стеке данные 32 бита и более выравниваются по границе двойного слова (.even)

Вызывающая функция освобождает стек от аргументов после вызова другой функции

Сохранение регистров

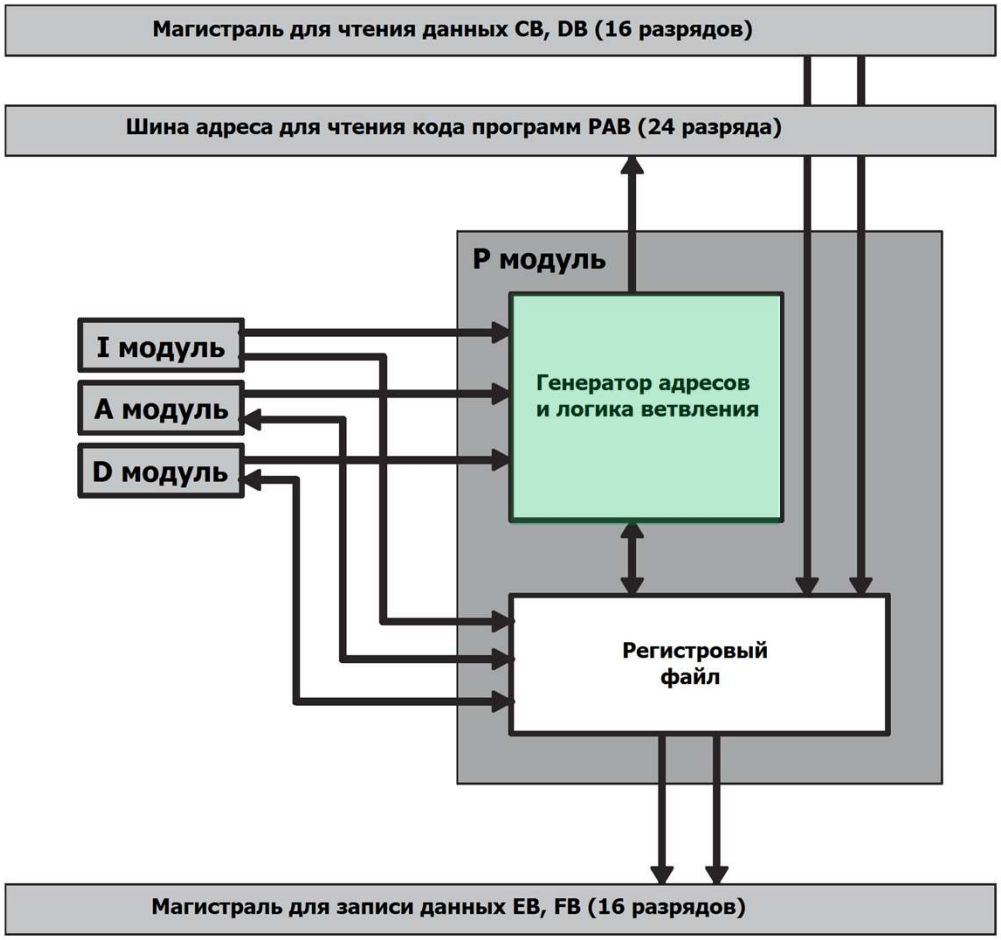
- **Вызываемая** функция:
  – должна сохранять и восстанавливать T2, T3, AR5-AR7 при использовании;
  – может использовать T0, T1 , AR0, AR1, …, AR4, AC0, AC1, …, AC3 без сохранения.
- **Вызывающая** функция должна сохранять перед вызовом используемые ей регистры T0, T1, AR0, AR1, …, AR4, AC0, AC1, …, AC3.
- При использовании сохраняются регистры RETA, BKx, BRCx, BRS1, BSAx, RSAx, REAx, RPTC, CSR, TRNx, (X)DP, (X)CDP, STx_55.

Локальные переменные

- При установленном флаге CPL в ST1 доступ к данным осуществляется через SP, в противном случае – через DP.
- Выделение (освобождение) локальной памяти: ASUB #n, SP (AADD #n, SP).


```
        // cdecl + fastcall
        …
        PSH     T2, AR7
        CALL    _func
        AADD    #4, SP
        …
_func:
        AADD    #-5, SP
        MOV     @6, T1      ; *SP(#6)
        MOV     #0, @1      ; *SP(#1)
        …
        AADD    #5, SP
        RET
```

Генератор адресов

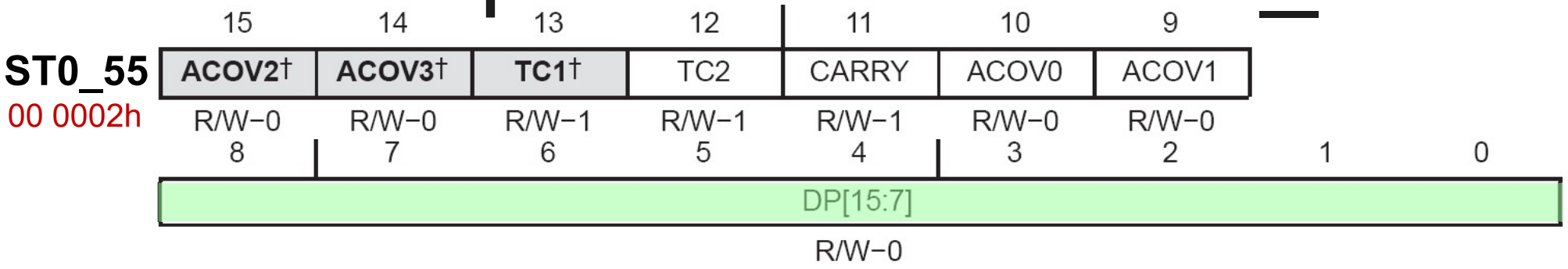


- **Генератор адресов команд**:
  – определение адреса следующей команды;
  – вычисление адреса команды, задаваемого командой ветвления;
  – формирование 24-битного адреса и его передача на шину адреса.

- **Логика ветвления**:
  – проверка флагов результата для команд ветвления (условных переходов);
  – передача результата проверки в генератор адресов;
  – инициализация обработки прерываний;
  – управление простым и блочным повторением команд;
  – управление параллельным выполнением команд.

- **Типы команд**:
  – обработки данных (D-Unit);
  – пересылка данных (A-Unit, D-Unit);
  – стековые операции (A-Unit);
  – ветвления (P-Unit);
  – циклического выполнения (P-Unit);
  – вызова подпрограмм (P-Unit);
  – обработки прерываний (P-Unit);
  – управление процессором (P-Unit).

Регистр состояния ST0_55



ACOV0-ACOV3 – **Ac**cumulator **Ov**erflow Flags **0-3**

(флаги переполнения регистров-аккумуляторов AC0-3, возникает при неравенстве переноса из знакового разряда переносу в знаковый разряд, фиксируется в 31 или 39 бите в зависимости от флага M40 из ST1_55, флаг переполнения сбрасывается командой очистки бит или командой условного ветвления).

CARRY – **Carry** bit

(флаг переноса за пределы разрядной сетки АЛУ, перенос фиксируется из 31 или 39 бита в зависимости от флага M40 из ST1_55, изменяется после выполнения арифметических и сдвиговых команд АЛУ).

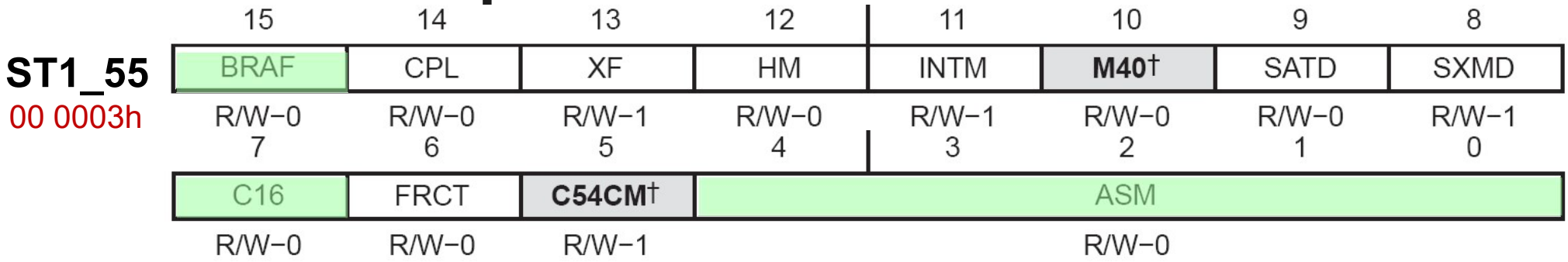
TC1-TC2 – **T**est/**C**ontrol Flag 1-2

(флаги расширения фиксируемых условий, изменяются некоторыми командами для использования в командах ветвления).

DP[15:7] – **D**ata **P**age bits at **15** to **7**

(копия 9-ти старших бит регистра страницы данных DP, используются для совместимости с младшими моделями микропроцессоров).

Регистр состояния ST1_55



C54CM – **C54**x **C**ompatible **M**ode
(флаг режима совместимости с моделью C54x).

ASM – **A**ccumulator **S**hift **M**ode (**только при C54CM=1**)
(число сдвигов аккумулятора, в C55x вместо поля ASM используются Tx).

BRAF – **B**lock **R**epeat **A**ctive **F**lag (**только при C54CM=1**)
(флаг активности повторения блока, в C55x вместо BRAF используется CFCT).

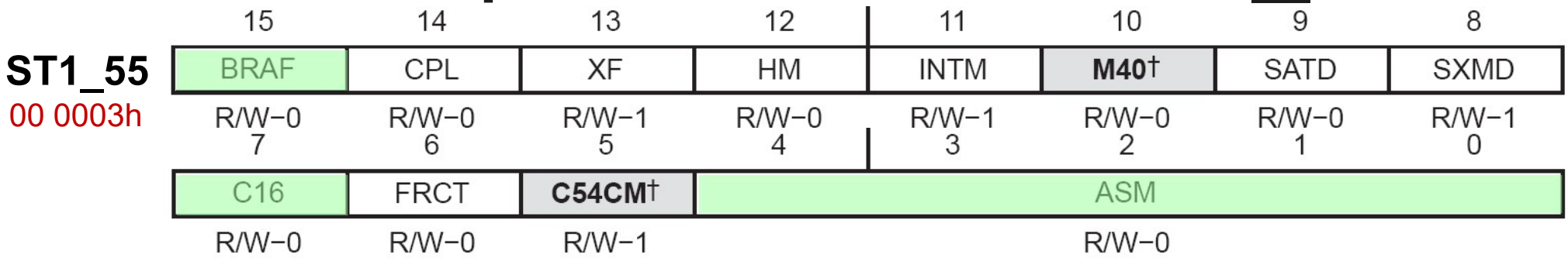
C16 – **C**ontrol **16** (**только при C54CM=1**)
(флаг режима разделения АЛУ на две независимые части).

CPL – **C**om**p**i**l**e Mode
(флаг режима прямой адресации компилятора: 0 – для адресации данных используется DP, 1 – для адресации данных используется SP; **MOV @2, T0**: MOV *DP(2), T0 или MOV *SP(2), T0).

XF – E**x**ternal **F**lag
(значение на внешнем выводе XF микропроцессора).

HM – **H**old **M**ode
(флаг режима ожидания внешнего интерфейса памяти EMIF: если EMIF в состоянии HOLD, то при HM=1 процессор ожидает, при 0 – продолжает работу).

Регистр состояния ST1_55



INTM – **Int**errupt **M**ode
(флаг глобального запрета маскируемых прерываний).

M40 – **M**ode **40**
(флаг режима работы операционного устройства: 0 – 32-битный режим, 1 – 40-битный режим).

SATD – **Sat**uration **D**-Unit
(флаг режима насыщения операционного устройства: 1 – выполняется насыщение до максимального 00 7FFF FFFFh или 7F FFFF FFFFh (минимального 00 8000 0000h или 80 0000 0000h) значения при переполнении; 0 – насыщение при переполнении не выполняется).

SXMD – **S**ign **E**xtension **Mo**de
(флаг режима расширения знака: 0 – расширение знака при загрузке регистров аккумулятора не выполняется, 1 – расширение знака при загрузке выполняется).

FRCT – **Fra**ctional Mode
(флаг дробного режима: FRCT=0 – результат умножения не сдвигается, FRCT=1 – результат умножения сдвигается влево на 1 разряд).

Регистр состояния ST2_55

ST2_55
00 004Bh

15	14	13	12	11	10	9	8
ARMS	Reserved		DBGM	EALLOW	RDM	Reserved	CDPLC
R/W−0	R−11b		R/W−1	R/W−0	R/W−0	R−0	R/W−0
7	6	5	4	3	2	1	0
AR7LC	AR6LC	AR5LC	AR4LC	AR3LC	AR2LC	AR1LC	AR0LC
R/W−0	R/W−0	R/W−0	R/W−0	R/W−0	R/W−0	R/W−0	R/W−0

AR0LC- AR7LC, **CDPLC**) – **AR0-AR7**, **CDPLC L**inear/**C**ircular
(флаги режима циклической адресации относительно AR0-AR7 и CDP).

RDM – **R**oun**d**ing **M**ode
(флаг режима округления: 0 – арифметическое округление, 1 – сигнальное (банковское) округление).

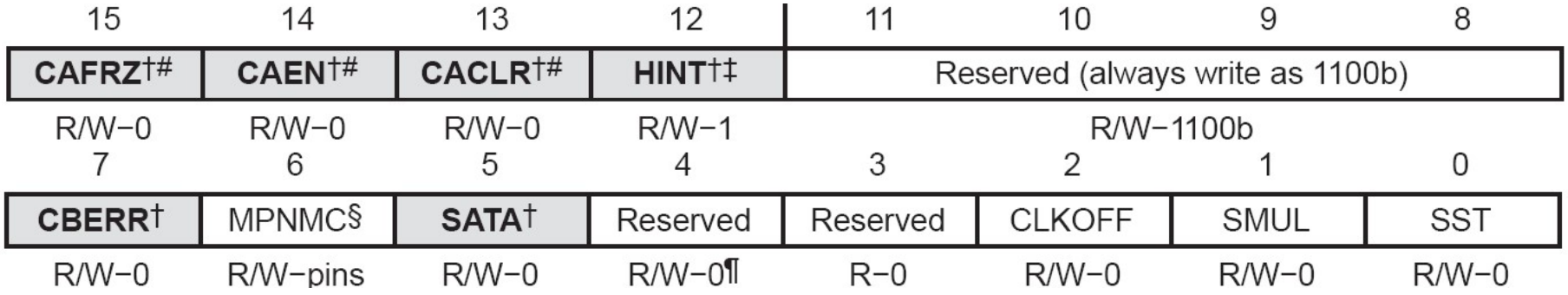
EALLOW – **E**mulation **Allow**
(флаг разрешения записи в регистры отладки – специальные регистры процессора, используемые при отладке программ).

DBGM – **D**e**b**u**g M**ode
(флаг режима откладки: 0 – режим отладки разрешен, возможна работа аппаратурного отладчика – эмулятора; 1 – режимы отладки запрещен, эмулятор отключен).

ARMS – **AR M**ode **S**witch
(переключатель режимов косвенной адресации: 0 – режимы косвенной адресации для интенсивной обработки сигнала, 1 – режимы косвенной адресации для приложений по управлению).

Регистр состояния ST3_55

ST3_55
00 0004h



CACLR – **Ca**sh **Cl**ea**r**
(флаг очистки кэша команд, сбрасывается процессором после очистки кэша).

CAEN – **Ca**sh **En**able
(флаг разрешения кэша команд).

CAFRZ – **Ca**sh **Fr**ee**z**e
(флаг замораживания кэша команд).

CBERR – **C**PU **B**us **Err**or
(флаг возникновения ошибки на шине процессора, вызывает установку флага прерывания BERRINTF, CBERR сбрасывается в процедуре прерывания).

HINT – **H**ost **Int**errupt
(флаг генерация прерывания ведущего процессора, подключенного через интерфейс EHPI - Enhanced Host-Port Interface).

MPNMC – **M**icro**p**rocessor/**M**icro**c**omputer
(флаг режима: MPNMC=0 – режим микрокомпьютера, при котором разрешается работа внутреннего ПЗУ в качестве памяти программ; MPNMC=1 – режим микропроцессора, при котором внутренне ПЗУ отключено).

Регистр состояния ST3_55

ST3_55
00 0004h

15	14	13	12	11	10	9	8
CAFRZ†#	CAEN†#	CACLR†#	HINT†‡	Reserved (always write as 1100b)			
R/W−0	R/W−0	R/W−0	R/W−1	R/W−1100b			
7	6	5	4	3	2	1	0
CBERR†	MPNMC§	SATA†	Reserved	Reserved	CLKOFF	SMUL	SST
R/W−0	R/W−pins	R/W−0	R/W−0¶	R−0	R/W−0	R/W−0	R/W−0

SATA – **Sat**uration **A**-Unit
(флаг режима насыщения адресного устройства: 1 – выполняется насыщение до максимального 7FFFh (минимального 8000h) значения при переполнении, 0 – насыщение при переполнении не выполняется).

SMUL – **S**aturation-on-**Mul**tiplication
(флаг режима насыщения при умножении: 1 – 18000h x 18000h насыщается до 7FFF FFFFh или 7F FFFF FFFFh; 0 – насыщение не выполняется).

CLKOFF – **Cl**oc**k Off**
(флаг запрета выдачи синхросигнала на выводе микропроцессора CLKOUT).

SST – **S**aturate-on-**St**ore (**только при C54CM=1**)
(флаг режима насыщения при сохранении аккумулятора в 32-битовый операнд: 1 – выполняется насыщение при сохранении; 0 – насыщение не выполняется; в C55x насыщение при сохранении кодируется в коде команды).