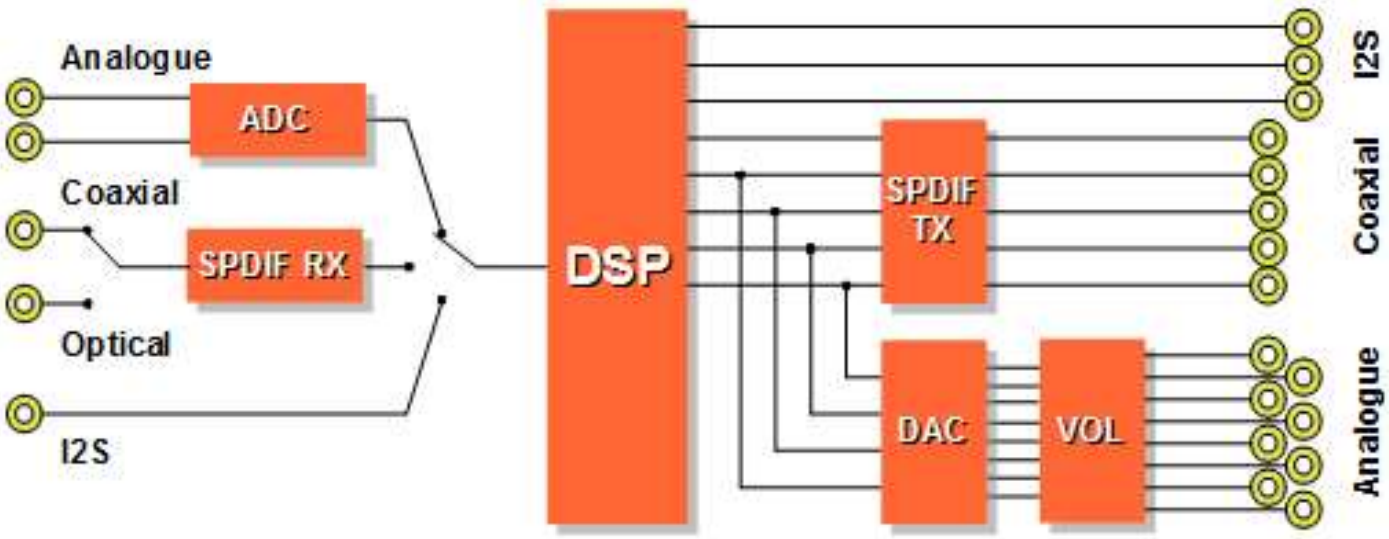
Микропроцессорные устройства обработки сигналов

Лекция L14
«Звуковой интерфейс»

http://vykhovanets.ru/course67/

Звуковые интерфейсы



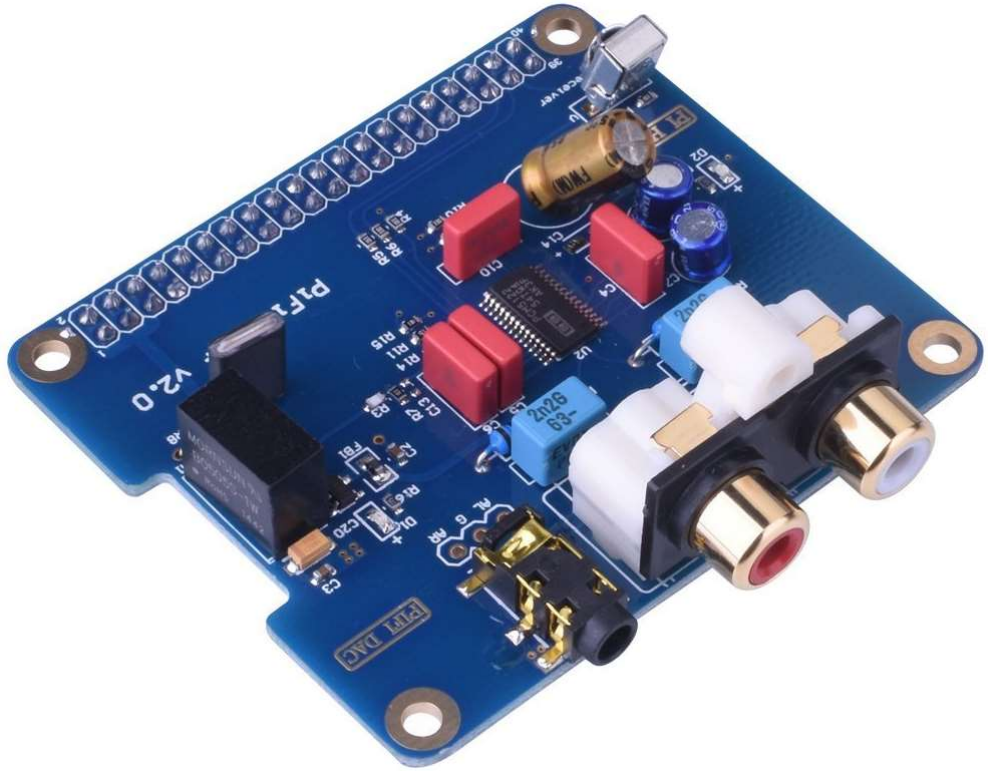
I²S – Integrated Inter-chip Sound (интегрированный межмикросхемный звук)

SPDIF – Sony-Philips Digital Interface

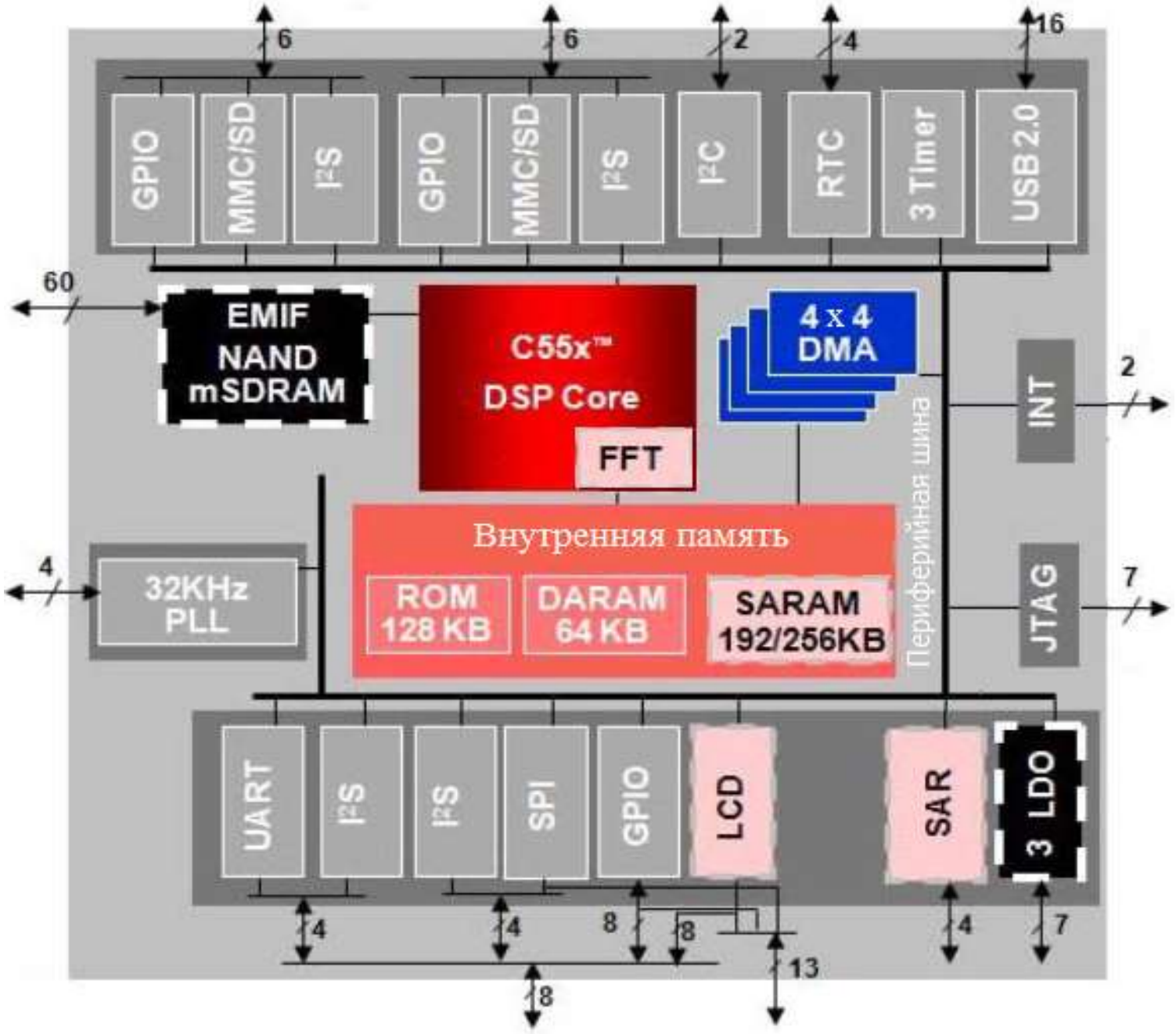
AES/EBU – Audio Engineering Society/European Broadcasting Union

Использование I2S

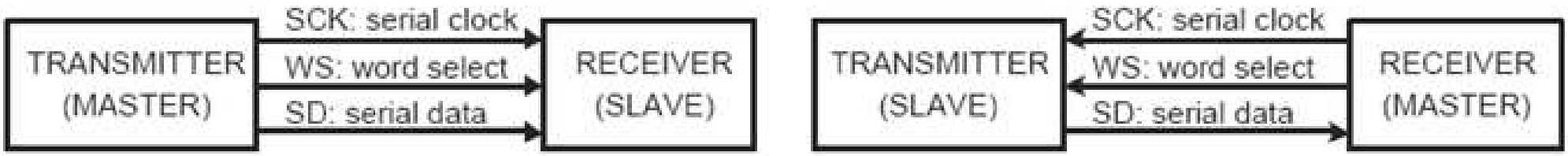
Звуковой кодек	CLK	I2S_CLK	**Контроллер I²S**
	WCLK	I2S_FS	
	DIN	I2S_DX	
	DOUT	I2S_RX	

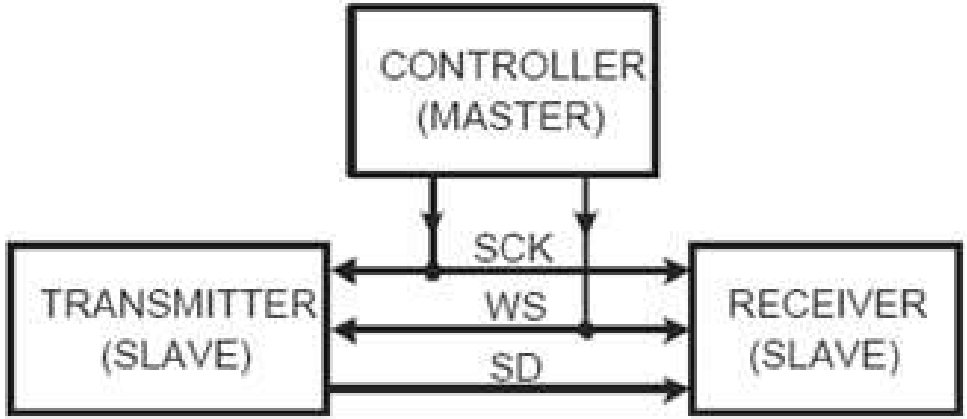


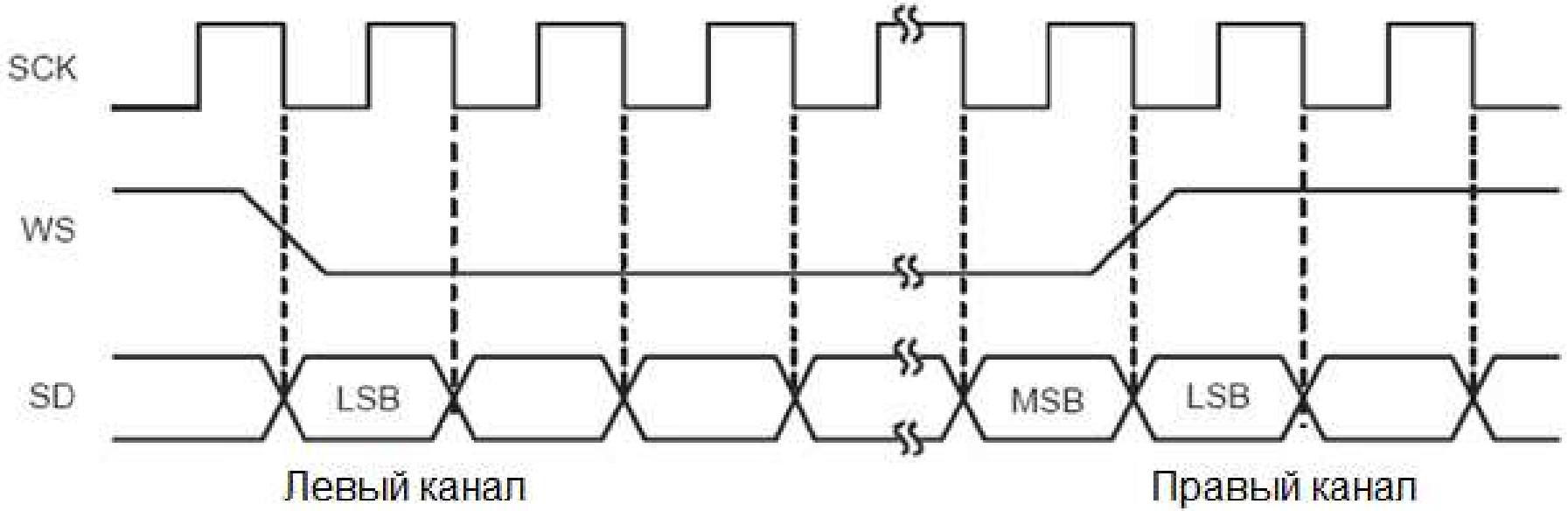
Состав микропроцессора



Подключение устройств I2S

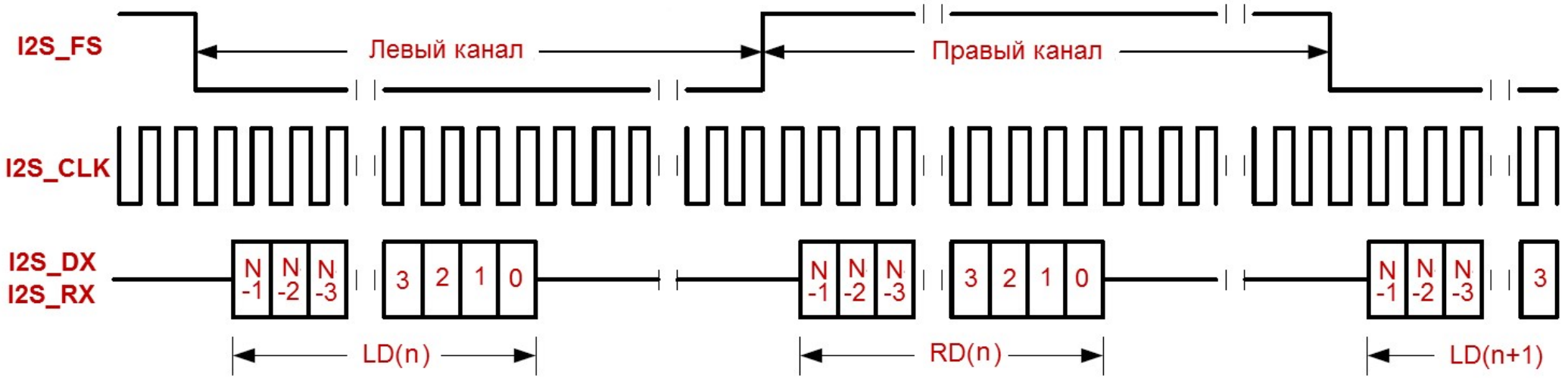




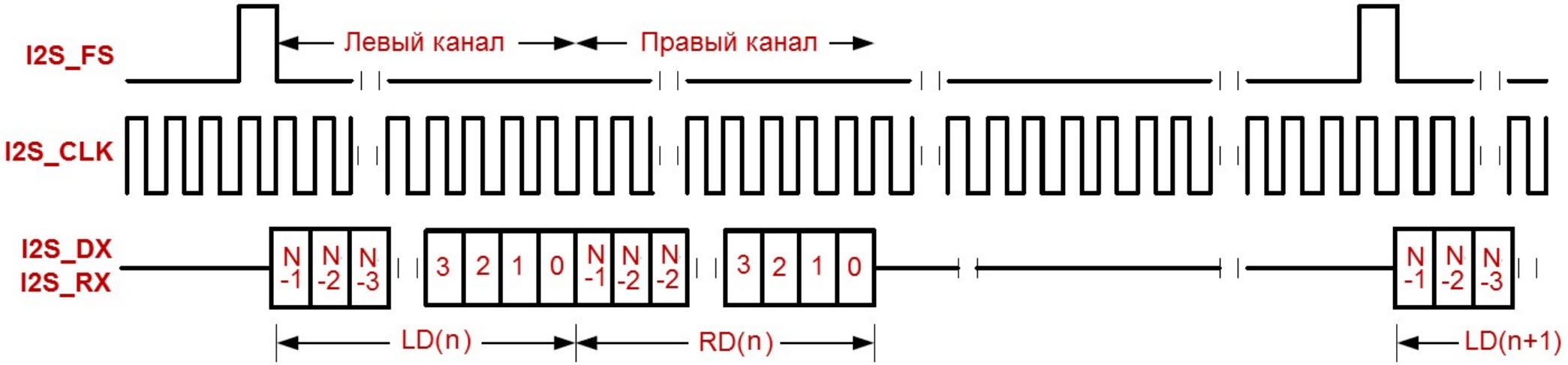


Форматы I2S

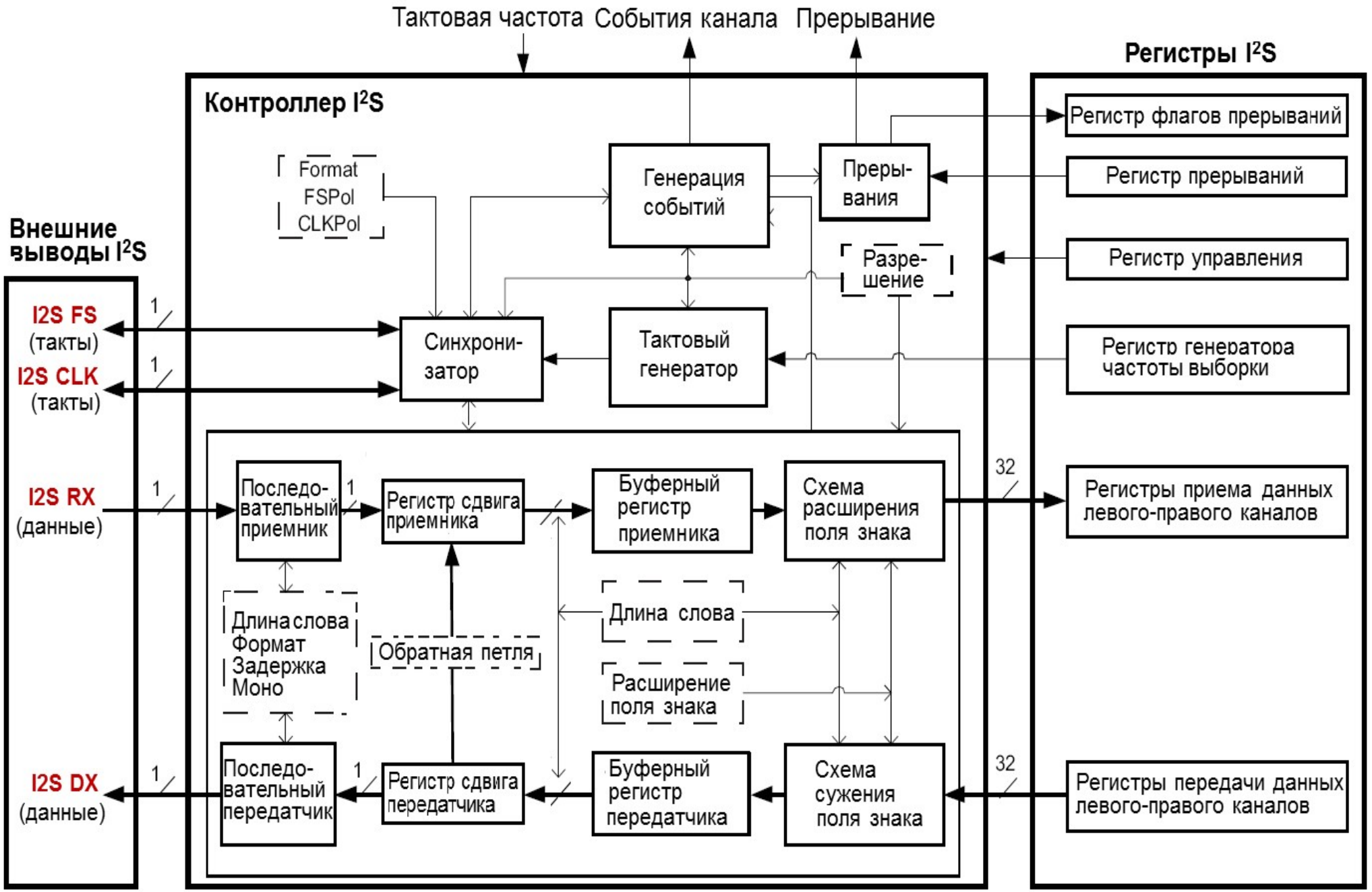
Формат I²S – Integrated Inter-chip Sound (интегрированный межмикросхемный звук)



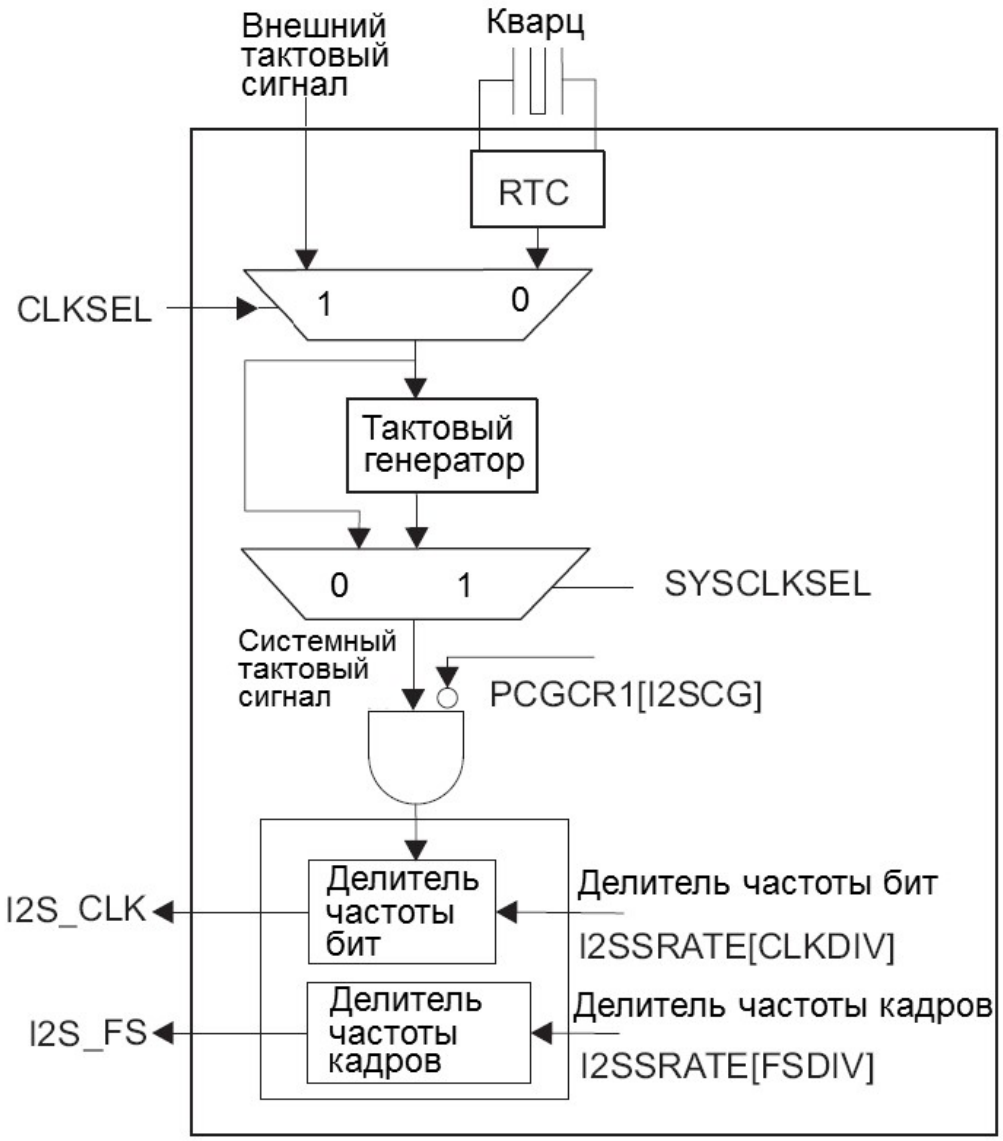
Формат DSP – Digital Signal Processing (цифровая обработка сигналов)



Организация контроллера I²S



Тактовое питание



$$f_{I2S\_CLK} = \frac{f_{SYSCLK}}{2^{CLKDIV+1}}$$

$$f_{I2S\_FS} = \frac{f_{I2S\_CLK}}{2^{FSDIV+1}}$$

PCGCR1 (1C02h) – Peripheral Clock Gating Configuration Register 1

15	14	13	12	11	10	9	8
SYSCLKDIS	I2S2CG	TMR2CG	TMR1CG	EMIFCG	TMR0CG	I2S1CG	I2S0CG
7	6	5	4	3	2	1	0
MMCSD1CG	I2CCG	Reserved	MMCSD0CG	DMA0CG	UARTCG	SPICG	I2S3CG

Регистры I²S

Базовые адреса: I2S0 – 2800h; I2S1 – 2900h; I2S2 – 2A00h; I2S3 – 2B00h.

Адрес	Обозначение	Описание
2800h	**I2SSCTRL**	Serializer Control Register (регистр управления)
2804h	**I2SSRATE**	Sample Rate Generator Register (регистр скорости выборки)
2808h	**I2STXLT0**	Transmit Left Data 0 Register (регистр передачи левый 0)
280Ch	**I2STXLT1**	Transmit Left Data 1 Register (регистр передачи левый 1)
2810h	**I2STXRT0**	Transmit Right Data 0 Register (регистр передачи правый 0)
2814h	**I2STXRT1**	Transmit Right Data 1 Register (регистр передачи правый 1)
2818h	**I2SINTFL**	Interrupt Flag Register (регистр флагов прерываний)
281Ch	**I2SINTMASK**	Interrupt Mask Register (регистр маски прерываний)
2820h	**I2SRXLT0**	Receive Left Data 0 Register (регистр приема левый 0)
2824h	**I2SRXLT1**	Receive Left Data 1 Register (регистр приема левый 1)
2828h	**I2SRXRT0**	Receive Right Data 0 Register (регистр приема правый 0)
282Ch	**I2SRXRT1**	Receive Right Data 1 Register (регистр приема правый 1)

PRCR (1C05h) – Peripheral Reset Control Register (регистр управления сбросом)

7	6	5	4	3	2	1	0
PG4_RST	Reserved	PG3_RST	DMA_RST	USB_RST	SAR_RST	PG1_RST	I2C_RST

PG4_RST: LCD, I2S2, I2S3, UART, SPI.
PG3_RST: MMC/SD0, MMC/SD1, I2S0, I2S1.

Запись нуля без эффекта, запись 1 запускает сброс, чтение 1 сигнализирует о состоянии сброса, чтение 0 – окончание сброса.

Вектор прерываний

Вектор		Прерывание	Приоритет	Адрес
00	RESET	Сброса и инициализации	00	IVPD:00h
01	NMI	Внутреннее немаскируемое	01	IVPD:08h
02	INT0	Внешнее по входу INT0	03	IVPD:10h
03	INT1	Внешнее по входу INT1	05	IVPD:18h
04	TINT	Агрегированное таймера	06	IVPD:20h
05	PROG0	I2S0, MMC/SD0 передачи	07	IVPD:28h
07	PROG1	I2S0,MMC/SD0 приема	10	IVPD:38h
08	DMA	Прямого доступа к памяти	11	IVPD:40h
09	PROG2	I2S1,MMC/SD1 передачи	13	IVPD:48h
11	PROG3	I2S1,MMC/SD1 приема	15	IVPD:58h
13	SAR	Агрегированное АЦП	18	IVPD:68h
14	XTM2	I2S2 передачи	21	IVPD:70h
15	RCV2	I2S2 приема	22	IVPD:78h
16	XMT3	I2S3 передачи	04	IVPH:80h
17	RCV3	I2S3 приема	09	IVPH:88h
18	RTC	Часов реального времени	12	IVPH:90h
21	GPIO	Портов ввода-вывода	20	IVPH:A8h
23	I2C	Контроллера I2C	24	IVPH:B8h

Регистр управления

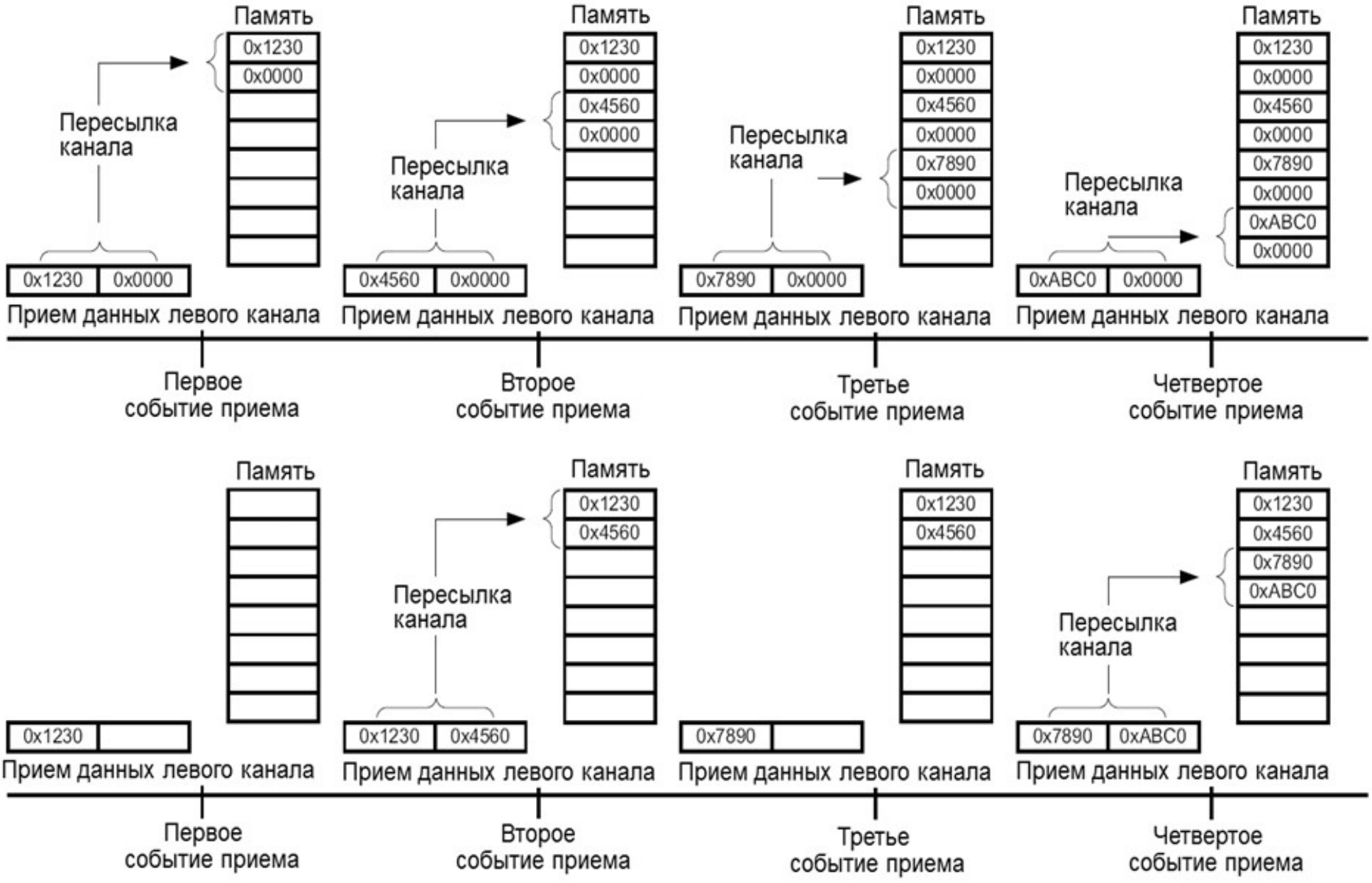
I2SSCTRL (2800h) – I2Sx Serializer Control Register (регистр управления)

15	14	13	12	11	10	9	8
ENABLE	Reserved		MONO	LOOPBACK	FSPOL	CLKPOL	DATADLY

7	6	5	2	1	0
PACK	SIGN_EXT	WDLNGTH		MODE	FRMT

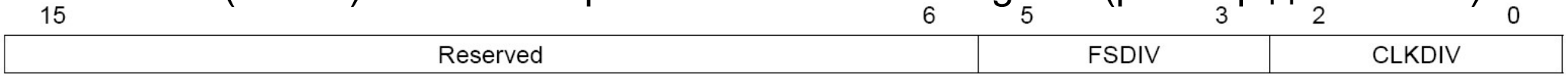
- **ENABLE** – Enable (разрешение работы контроллера)
- **MONO** – Mono (режим моно)
- **LOOPBACK** – Loopback (режим обратной петли)
- **FSPOL** – Frame-synchronization polarity (полярность кадровой синхронизации I2S_FS: 0 – высокий уровень, 1 – низкий уровень)
- **CLKPOL** – Clock polarity (полярность битовой синхронизации I2S_CLK: 0 – переход от 0 к 1, 1 – переход от 1 к 0)
- **DATADLY** – Data delay (задержка данных: 0 – 1 бит, 1 – 2 бита)
- **PACK** – Pack (упаковка данных: 0 – запрещена, 1 – разрешена)
- **SIGN_EXT** – Sign extension (расширение знака)
- **WDLNGTH** – Word length(длина слова: 0 – 8 бит, 1 – 10 бит, 2 – 12 бит, 3 – 14 бит, 4 – 16 бит, 5 – 18 бит, 6 – 20 бит, 7 – 24 бита, 8 – 32 бита)
- **MODE** – Mode (режим: 1 – ведущий, master; 0 – ведомый, slave)
- **FRMT** – Format (формат: 0 – I2S, 1 – DSP)

Упаковка данных



Регистры делителей и данных

I2SSRATE (2804h) – I2Sn Sample Rate Generator Register (регистр делителей)



I2STXLT0 (2808h) – I2Sx Transmit Left Data 0 Register (регистр передачи левый 0)
I2STXLT1 (280Ch) – I2Sx Transmit Left Data 1 Register (регистр передачи левый 1)
I2STXRT0 (2810h) – I2Sx Transmit Right Data 0 Register (регистр передачи правый 0)
I2STXRT1 (2814h) – I2Sx Transmit Right Data 1 Register (регистр передачи правый 1)
I2SRXLT0 (2820h) – I2Sx Receive Left Data 0 Register (регистр передачи левый 0)
I2SRXLT1 (2824h) – I2Sx Receive Left Data 1 Register (регистр передачи левый 1)
I2SRXRT0 (2828h) – I2Sx Receive Right Data 0 Register (регистр передачи правый 0)
I2SRXRT1 (282Ch) – I2Sx Receive Right Data 1 Register (регистр передачи правый1)

- **FSDIV** – Frame-synchronization divider (делитель кадровой синхронизации: 0 – 8, 1 – 16, 2 – 32, 3 – 64, 4 – 128, 5 – 256)
- **CLKDIV** – Clock divider (делитель битовой синхронизации: 0 – 2, 1 – 4, 2 – 8, 3 – 16, 4 – 32, 5 – 64, 6 – 128, 7 – 256)
- **DATA** – Data (данные)

Регистры прерываний

I2SINTFL (2818h) – I2Sx Interrupt Flag Register (регистр флагов прерываний)
I2SINTMASK (281Ch) – I2Sx Interrupt Mask Register (регистр масок прерываний)

15 … 8	7	6	5	4	3	2	1	0
Reserved	Reserved		XMITSTFL	XMITMONFL	RCVSTFL	RCVMONFL	FERRFL	OUERR

- **XMITSTFL** – Stereo data transmit (флаг или маска прерывания по готовности стерео передатчика)
- **XMITMONFL** – Mono data transmit (флаг или маска прерывания по готовности моно передатчика)
- **RCVSTFL** – Stereo data receive (флаг или маска прерывания по готовности стерео приемника)
- **RCVMONFL** – Mono data receive (флаг или маска прерывания по готовности моно приемника)
- **FERRFL** – Frame-synchronization error (флаг или маска прерывания по ошибки кадровой синхронизации)
- **OUERRFL** – Overrun or Underrun condition (флаг или маска прерывания по передержке или недодержке)

Инициализация контроллера

- Сброс PRCR[PGx_RST] и ожидание нуля.
- Подача тактового питания PCGCR1[I2SxCG]=0.
- Сброс канала прямого доступа*.
- Запрет прерываний и очистка флагов I2Sx.
- Инициализация канала прямого доступа*.
- Конфигурирование внешнего устройства I2S.
- Разрешение прерываний.
- Конфигурирование контроллера I2S:
  – трассировка сигналов I2S на внешние входы;
  – если I2S ведущее устройство, то задать тактовую частоту в I2SSRATE;
  – разрешить прерывания в I2SINTMASK;
  – сконфигурировать I2SSCTRL.
- Обработка прерываний контроллера и канала*

Обработчик прерываний

- Чтение I2SINTFL для сброса флагов прерываний.
- Чтение или запись данных в регистры данных I2S, если не используется канал прямого доступа.
- Возврат из прерывания.

Завершение ввода-вывода

- Запрет прерываний и очистка флагов I2Sx.
- Останов канала прямого доступа*.
- Снятие тактового питания PCGCR1[I2SxCG]=0.

Функции I²S

```
      .text
      .global _port_read
      .global _port_write

; int port_read(int reg)
_port_read:
      MOV  T0, AR0
      MOV  port(*AR0), T0
      ret

; void port_write(int reg, int data)
_port_write:
      MOV  T0, AR0
      MOV  T1, port(*AR0)
      ret
```

```
void i2s2_init(void)
{
   int tmp = 0x04<<2; // WORD 16
   tmp |= 0x0082;  // Master, Pack
   port_write(0x2A00, tmp);  // 16 bit
   tmp = port_read(0x2A00); // I2SSCTRL

   tmp = 0x2<<3; // FS 32;
   tmp |= 0x0005;  // Clock = CPU / 4
   port_write(0x2A04, tmp); // I2SSRATE
   port_write(0x2A1C, 0x20);// I2SINTMASK
}
```

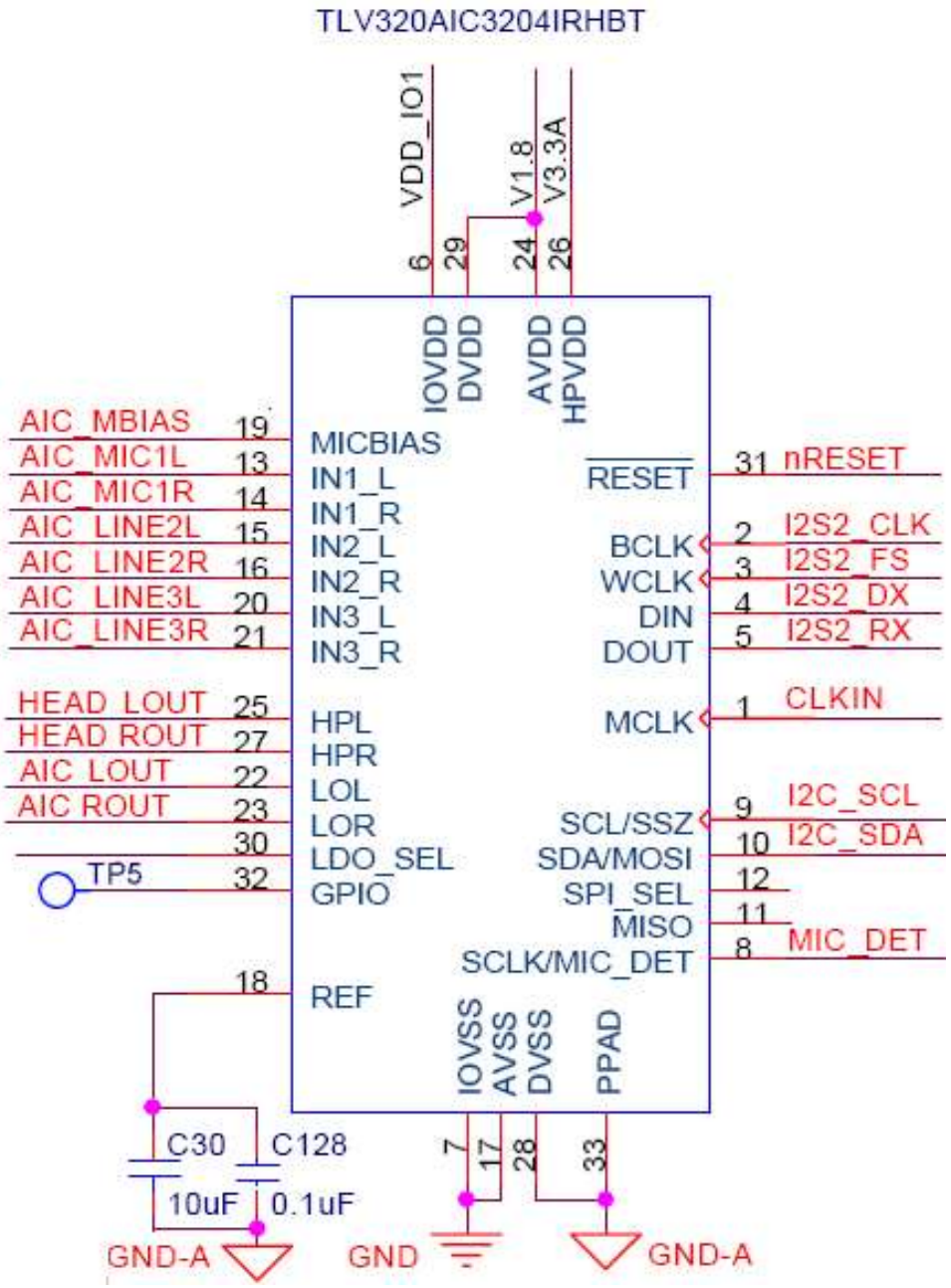
```
port_write(0x2A08, 0x5678); // I2STXLT0
port_write(0x2A0C, 0x1234); // I2STXLT1
port_write(0x2A10, 0x5678); // I2STXRT0
port_write(0x2A14, 0x1234); // I2STXRT1
```

Определения I²S

```
#define I2S2_CR             *(volatile ioport Uint16*)(0x2A00)
#define I2S2_SRGR           *(volatile ioport Uint16*)(0x2A04)
#define I2S2_W0_LSW_W       *(volatile ioport Uint16*)(0x2A08)
#define I2S2_W0_MSW_W       *(volatile ioport Uint16*)(0x2A09)
#define I2S2_W1_LSW_W       *(volatile ioport Uint16*)(0x2A0C)
#define I2S2_W1_MSW_W       *(volatile ioport Uint16*)(0x2A0D)
#define I2S2_IR             *(volatile ioport Uint16*)(0x2A10)
#define I2S2_ICMR           *(volatile ioport Uint16*)(0x2A14)
#define I2S2_W0_LSW_R       *(volatile ioport Uint16*)(0x2A28)
#define I2S2_W0_MSW_R       *(volatile ioport Uint16*)(0x2A29)
#define I2S2_W1_LSW_R       *(volatile ioport Uint16*)(0x2A2C)
#define I2S2_W1_MSW_R       *(volatile ioport Uint16*)(0x2A2D)
```

Инициализация

```
/* Initialize CPU */
c5515_init();
/* Configure Parallel Port */
SYS_EXBUSSEL &= ~0x7000;
SYS_EXBUSSEL |= 0x1000;  // Configure Parallel
/* Configure Serial Port */
SYS_EXBUSSEL &= ~0x0C00;
SYS_EXBUSSEL |= 0x0400; // Serial Port mode 1 (
c5515_GPIO_init();
c5515_GPIO_setDirection(GPIO10, GPIO_OUT);
c5515_GPIO_setOutput(GPIO10, 1); // Take AIC320
/* Initialize I2C */
I2C_init();
/* Interrupts */
asm(" BSET INTM");
unsigned long int vector;
vector = (unsigned long int) &VECSTART;
vector = vector >> 8;
IVPD = (unsigned short) vector;
IVPH = (unsigned short) vector;
IER0 |= 0x4000;
asm(" BCLR INTM");
/* I2S settings */
I2S2_SRGR = 0x0015;
I2S2_ICMR = 0x0028; // Enable interrupts
I2S2_CR = 0x8012;   // 16-bit word, Master, ena
/* Aution codec */
aic3204_stereo_in1();
/* ... */
```



Секция прерываний

```
    .sect "vectors"
    .global _VECSTART
    .global _i2s2_tx_isr
    .global _i2s2_rx_isr
    .global _Reset
    .ref _c_int00
_VECSTART:
_Reset  .ivec _c_int00, USE_RETA
nmi     .ivec no_isr
int0    .ivec no_isr
; ...
int11   .ivec no_isr
int12   .ivec _i2s2_tx_isr
int13   .ivec no_isr
; ...
int29   .ivec no_isr
        .text
        .def no_isr
no_isr: b #no_isr
```

```
MEMORY
{
PAGE 0:
MMR (RWIX): origin = 0x000000, length = 0x0000c0
DARAM0 (RWIX): origin = 0x0000c0, length = 0x00ff40
SARAM0 (RWIX): origin = 0x010000, length = 0x010000
SARAM1 (RWIX): origin = 0x020000, length = 0x020000
SARAM2 (RWIX): origin = 0x040000, length = 0x00FE00
VECS (RWIX): origin = 0x04FE00, length = 0x000200
PDROM (RIX): origin = 0xff8000, length = 0x008000
PAGE 2:
IOPORT (RWI) : origin = 0x000000, length = 0x020000
}
SECTIONS
{
.text >> SARAM1|SARAM2|SARAM0
.stack > DARAM0
.sysstack > DARAM0
.data >> DARAM0|SARAM0|SARAM1
.bss >> DARAM0|SARAM0|SARAM1
.const >> DARAM0|SARAM0|SARAM1
.sysmem > DARAM0|SARAM0|SARAM1
.switch > SARAM2
.cinit > SARAM2
.pinit > SARAM2
.cio > SARAM2
.args > SARAM2
vectors > VECS
.ioport > IOPORT PAGE 2
}
```

Обработчик прерываний I²S

```
interrupt void i2s2_tx_isr()
{
    I2S2_W0_MSW_W = buf_out_left_2[buf_index];
    I2S2_W1_MSW_W = buf_out_right_2[buf_index];

    while ((Rcv & I2S2_IR) == 0);

    buf_in_left_2[buf_index] = I2S2_W0_MSW_R;
    buf_in_right_2[buf_index] = I2S2_W1_MSW_R;

    buf_index++;

    if (buf_index >= W_LEN) {
        change = buf_in_left_2;
        buf_in_left_2 = buf_in_left_1;
        buf_in_left_1 = change;

        change = buf_out_left_2;
        buf_out_left_2 = buf_out_left_1;
        buf_out_left_1 = change;

        change = buf_in_right_2;
        buf_in_right_2 = buf_in_right_1;
        buf_in_right_1 = change;

        change = buf_out_right_2;
        buf_out_right_2 = buf_out_right_1;
        buf_out_right_1 = change;

        buf_index = 0;
        effect_flag = EFFECT_FLAG;
    }
}
```

```
Int16 b1[W_LEN] = { 0 };
Int16 b2[W_LEN] = { 0 };
Int16 b3[W_LEN] = { 0 };
Int16 b4[W_LEN] = { 0 };

Int16 b5[W_LEN] = { 0 };
Int16 b6[W_LEN] = { 0 };
Int16 b7[W_LEN] = { 0 };
Int16 b8[W_LEN] = { 0 };

Int16* buf_in_left_1 = b1;
Int16* buf_in_right_1 = b2;
Int16* buf_out_left_1 = b3;
Int16* buf_out_right_1 = b4;

Int16* buf_in_left_2 = b5;
Int16* buf_in_right_2 = b6;
Int16* buf_out_left_2 = b7;
Int16* buf_out_right_2 = b8;
```