Последовательный канал информационного обмена по стандарту ARINC-429 (ГОСТ 18977-79)
Лабораторная работа №405DN

В данной работе осваиваются методы проектирования цифровых модулей передатчика и приемника сигналов стандарта ARINC 429.

Данные по стандарту ARINC 429 передаются последовательно словами по 32 бита. Скорость передачи 12.5, 50 или 100 кбит/сек. Логической 1 соответствует положительный а, логическому 0 – отрицательный импульс. Длительность импульса равна половине интервала следования (длительности такта). Структура ARINC слова приведена в таблице 1. Первые 8 бит являются адресом абонента, следующие 23 бита являются данными, а последний контрольный бит дополняет число единиц в слове до нечетного числа. Особенностью формата ARINC слова является то, что первые 8 бит адреса передаются старшими битами вперед, а 23 бита данных передаются младшими битами вперед. Инверсия нумерации бит считается недопустимой. Минимальная пауза между словами 4 бита.

Таблица 1

№ бита	Имя	Назначение
0	ADR[7]	Адрес абонента (старшими битами вперед)
1	ADR[6]	
2	ADR[5]	
3	ADR[4]	
4	ADR[3]	
5	ADR[2]	
6	ADR[1]	
7	ADR[0]	
8	DAT[0]	DAT[7:0] Младший байт данных (младшими битами вперед)
9	DAT[1]	
10	DAT[2]	
11	DAT[3]	
12	DAT[4]	
13	DAT[5]	
14	DAT[6]	
15	DAT[7]	
16	DAT[8]	DAT[15:8] Средний байт данных (младшими битами вперед)
17	DAT[9]	
18	DAT[10]	
19	DAT[11]	
20	DAT[12]	
21	DAT[13]	
22	DAT[14]	
23	DAT[15]	
34	DAT[16]	DAT[22:16] Старший байт данных (младшими битами вперед)
25	DAT[17]	
26	DAT[18]	
27	DAT[19]	
28	DAT[20]	
29	DAT[21]	
30	DAT[22]	
31	CB	Контрольный бит (дополнение до нечетного числа 1)

На логическом уровне передатчик должен иметь 2 выхода:

Tx1 – канал 1,
Tx0 – канал 0.

Импульсы каналов не должны пересекаться и иметь длительность T_{pulse} равной половине длительности такта ($T_{pulse} = T_{bit}/2 = 1/(2*VEL)$).

Ниже приведена схема модуля **AR_TXD** ARINC передатчика, написанная на лаконичном диалекте языка VERILOG.

```
module AR_TXD (
                input clk,              output wire ce,          // Скорость (Tce=1/Vel)
                input [1:0] Nvel,       output wire Tx1,         //Импульсы канала 1
                input [7:0] adr,        output wire Tx0,          //Импульсы канала 0
                input [22:0]dat,        output wire SLP,         // Крутизна фронтов
                input st,               output reg en_tx_dat=0,// Интервал передачи данных
                                        output wire T_end_dat,// Такт конца кадра
                                        output reg ft_tx=0,       // Триггер контроля четности
                                        output wire txd,         // Последовательные данные
                                        output reg qm=0,         // Модулятор
                                        output reg[5:0]cb_bit=0,//Счетчик бит
                                        output reg en_tx_word=0);//Интервал передачи слова

parameter Fclk=50000000 ;  //50 MHz
parameter V1Mb=1000000 ; // 1000 kb/s
parameter V100kb=100000 ; // 100 kb/s
parameter V50kb= 50000 ;    // 50 kb/s
parameter V12_5kb=12500 ; // 12.5 kb/s

wire [10:0]AR_Nt= (Nvel [1:0]==3)? (Fclk/(2*V1Mb)) : //1000.000 kb/s
                                    (Nvel [1:0]==2)? (Fclk/(2*V100kb)) : // 100.000 kb/s
                                    (Nvel [1:0]==1)? (Fclk/(2*V50kb)) : // 50.000 kb/s
                                                        (Fclk/(2*V12_5kb)) ; // 12.500 kb/s

reg [10:0]cb_tce=0 ;              // Счетчик такта
reg [ 7:0]sr_adr=0 ;              // Регистр сдвига адреса
reg [22:0]sr_dat=0 ;              // Регистр сдвига данных
wire ce_tact = (cb_tce==AR_Nt); // Tce_tact=1/(2*VEL)
assign ce = ce_tact & qm ;      // Tce=1/VEL
assign T_end_dat = (cb_bit==31); // Такт контрольного бита
assign ce_end_dat = T_end_dat & ce ; //Импульс конца данных
assign ce_end_word = (cb_bit==35) & ce ;//Импульс конца слова
assign txd = sr_adr[7] | (T_end_dat & ft_tx) ;//Последовательные данные
assign Tx1 = en_tx_dat & qm & txd ; //Импульсы канала 1
assign Tx0 = en_tx_dat & qm & !txd ; //Импульсы канала 0
assign SLP = (Nvel ==0) ; // Крутизна фронтов
wire start = st & !en_tx_word ; // Запрет запуска пока идет передача слова

always @ (posedge clk) begin
cb_tce <= (start | ce_tact)? 1 : cb_tce+1 ;
qm <= start? 0 : (en_tx_word & ce_tact)? !qm : qm ;//Переключение триггера модулятора
cb_bit <= start? 0 : (en_tx_word & ce)? cb_bit+1 : cb_bit ;//Счет бит
en_tx_word <= start? 1 : ce_end_word? 0 : en_tx_word ; //Формирование интервала передачи с паузой
```

```
en_tx_dat <= start? 1 : ce_end_dat? 0 : en_tx_dat ;//Формирование интервала передачи данных
ft_tx <=(start | ce_end_dat)? 1:(sr_adr[7] & (ce & en_tx_dat & !T_end_dat))? !ft_tx: ft_tx ; //Счет четности
sr_adr <= start? adr : (ce & en_tx_dat)? sr_adr <<1 | sr_dat[0] : sr_adr ; //Сдвиг адреса старшими битами вперед
sr_dat <= start? dat : (ce & en_tx_dat)? sr_dat >>1 : sr_dat ; //Сдвиг данных младшими битами вперед
end
endmodule
```

Входами модуля являются:

clk – сигнал синхронизации с периодом Tclk=20 нс,

Nvel[1:0] – указатель скорости,

adr[7:0] – 8 бит адреса,

dat[22:0] – 23 бита данных,

st – положительный импульс запуска с длительностью Tclk.

Указатель скорости имеет 4 состояния, 4-е состояние Nvel[1:0]=3 обеспечивает не стандартную скорость 1000 кбит/сек, которую удобно использовать при моделировании и отладке схемы модуля.

Кроме необходимых сигналов Tx0 и Tx1 на выходные порты модуля выведены дополнительные сигналы, которые в данной работе предназначены для визуального контроля процесса работы внутренних элементов модуля при моделировании.

Сигнал SLP предназначен для управления крутизной фронтов сигналов на линии передачи. Например, микросхема HI-8570 (рис.1) фирмы HOLT, специализирующейся на выпуске микросхем для ARINC-429, имеет вход SLP1.5/10, который управляет длительностью фронтов выходных сигналов. При SLP=0 длительность фронтов импульсов равна 1.5 мкс, а при SLP=1 длительность фронтов импульсов равна 10 мкс.

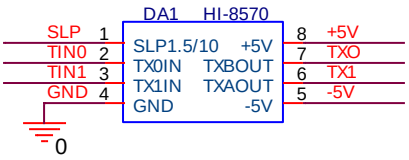

Рис.1 Микросхема выходного драйвера ARINC-429 фирмы HOLT

1. Задание к допуску

Получить от преподавателя номер набора параметров (Таблица 2), в который входят: скорость VEL (кбит/сек) адрес ADR[7:0] и данные DAT[22:0].

Таблица 2

№	Vel (kbit/sec)	ADR[7:0] (HEX,bin)	DAT[22:0] (HEX)
1	100	8'h81=8'b10000001	23'h123400
2	50	8'h82=8'b10000010	23'h567800
3	12.5	8'h83=8'b10000011	23'h789A00
4	100	8'h84=8'b10000100	23'h112200
5	50	8'h85=8'b10000101	23'h2A5500
6	12.5	8'h86=8'b10000110	23'h448800

7	50	8’h87=8’b10000111	23’h6E1100
8	100	8’h88=8’b10001000	23’h4C6600
9	12.5	8’h89=8’b10001001	23’h087700
10	100	8’h8A=8’b10001010	23’h1ABC00
11	50	8’h8B=8’b10001011	23’h32C300
12	12.5	8’h8C=8’b10001100	23’h26A500
13	50	8’h8D=8’b10001101	23’h702D00
14	100	8’h8E=8’b10001110	23’h70E100
15	50	8’h8F=8’b10001111	23’h633C00
16	12.5	8’h90=8’b10010000	23’h3C4D00
17	50	8’h91=8’b10010001	23’h702D00
18	100	8’h92=8’b10010010	23’h70E100
19	50	8’h93=8’b10010011	23’h433C00
20	12.5	8’h94=8’b10010100	23’h3C4D00

Старший бит адреса у всех вариантов равен 1, а число образованное младшими битами совпадает с номером варианта. Единица старшего бита адреса позволяет отображать все слово на экране осциллографа в режиме ждущей развертки с запуском фронтом первого импульса Tx1 канала единиц.

1.1 Переписать в тетрадь схему модуля **AR_TXD.** Прочитать и записать все комментарии к регистрам и сигналам.

1.1.2 Начертить в тетради эскиз временных диаграмм генератора **AR_TXD** сигналов Tx1 и Tx0 для 8 бит адреса заданного варианта параметров.

2. Задание к выполнению

Создать проект с именем Lab405D, для ПЛИС, используемой в макете NEXYS-2.

2.1 В окне источников (Sources) создать (New Source) модуль AR_TXD генератора сигналов Tx1 и Tx0 стандарта ARINC 429 и далее на Verilog-е или в «схематике» составить схему этого модуля. Выполнить синтез (Sinthesize - XST) созданного модуля. При необходимости исправить синтаксические ошибки.

Создать для этого модуля задание на моделирование (Verilog Test Fixture). Для входных сигналов установить заданные значения (см. таблицу 2).

2.1.1 Провести моделирование работы модуля AR_TXD при заданных параметрах. Проверить правильность формирования контрольного бита. Зарисовать в тетради, полученные временные диаграммы сигналов.

2.1.2 Проверить возможность повторного запуска до полного окончания передачи слова.

2.2 Предложить алгоритм приема ARINC слова. На основе предложенного алгоритма составить схему модуля AR_RXD приемника сигналов Tx1 и Tx0 модуля AR_TXD. Обязательными выводами окончательного варианта отлаженного модуля AR_RXD должны быть входы: In1, In0 и выходы: регистр адреса, регистр данных и сигнал подтверждения правильности приема слова. При отладке могут потребоваться и другие выводы, поэтому при моделировании работы приемника AR_RXD более удобно использовать текстовое описание связей между модулями. «Текстовая» схема избавляет от необходимости редактировать или создавать новый символ при добавлении или удалении выводов.
Пример такой схемы приведен в приложении 4.1.

2.3 Начертить в тетради временные диаграммы входных и выходных сигналов отлаженного модуля AR_RXD.

3. Задание к сдаче работы

3.1 Составить схему Sch_Lab405D. Пример такой схемы приведен на рис.2. В состав схемы должны входить:

- модуль передатчика AR_TXD,
- модуль источника адреса и данных ADR_DAT_BL,
- модуль приемника AR_RXD,
- модуль регистра записи принятых адреса и данных AR_FD32,
- модуль семи сегментного индикатора DISPLAY для отображения передаваемых и принятыхых данных,
- модуль мультиплексора AR_MUX передаваемых и принятых данных для DISPLAY/

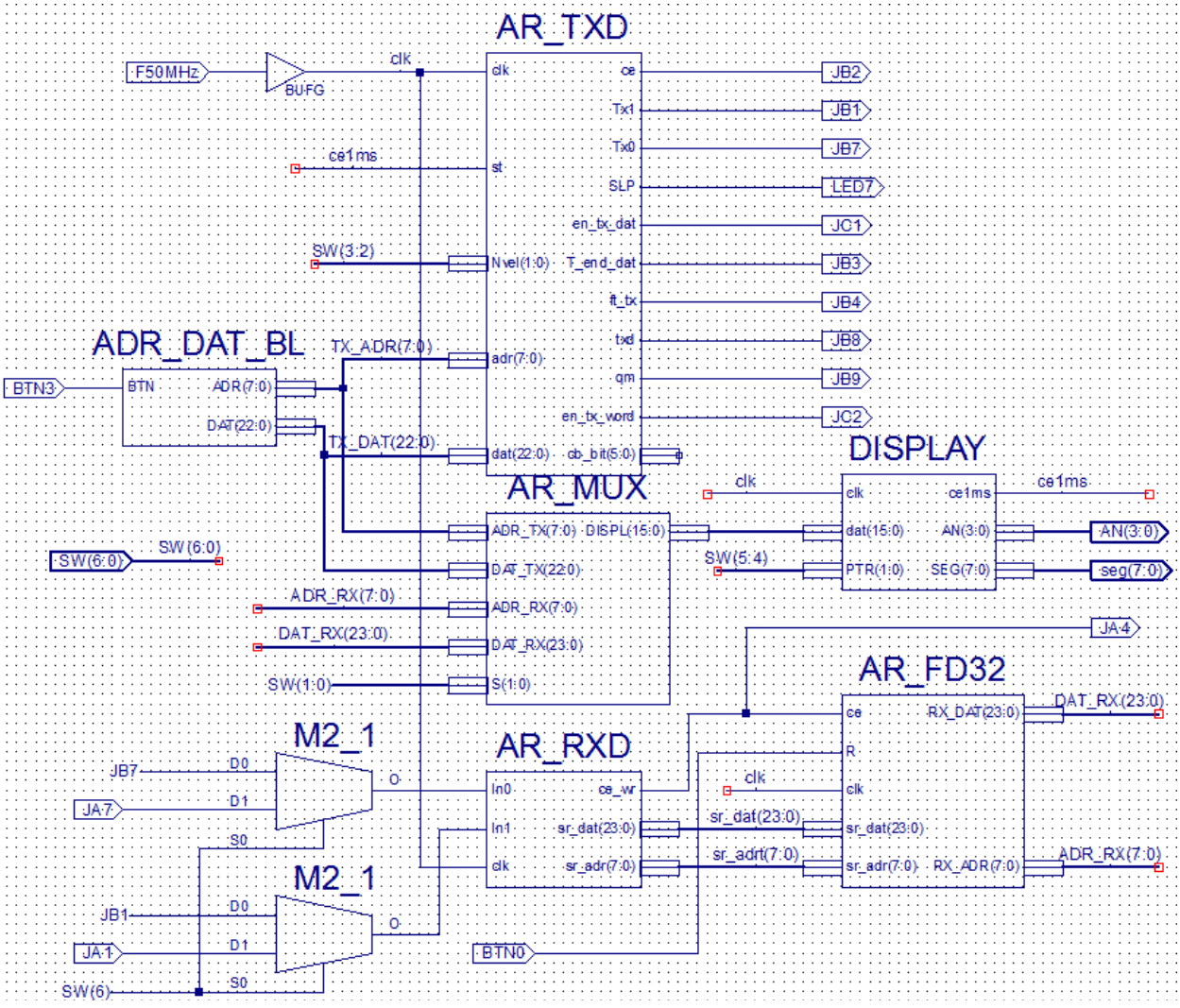

Рис.2 Пример схемы для сдачи работы

Модуль источника ADR_DAT_BL имеет вход BTN, логическая единица на котором инвертирует один бит данных, например, бит, номер которого совпадает с номером варианта. Это позволяет наглядно показывать реакцию контрольного бита на число единиц в слове.

Пример выделения модулем AR_MUX 16 бит данных для модуля DISPLAY из 32 бит передаваемых и принятых ARINC слов передатчика и приемника:

- {DAT_TX[7:0],ADR_TX[7:0]},

 {1'b0,DAT_TX[22:8]},
 {DAT_RX[7:0],ADR_RX[7:0]},
 {DAT_RX[23:8]}.

Входные мультиплексоры M2_1 при SW(6)=0 соединяют входы In0 и In1 приемника с выходами Tx0 и Tx1 передатчика без внешних проводных связей (JB1->JA1, JB7->JA7). Кнопка BTN0 «сбрасывает» принятые данные.

3.2 Выполнить синтез и имплементацию схемы, загрузить в ПЛИС макета продемонстрировать работу схемы.

4. Приложения

4.1 Пример схемы тестирования модуля приемника

```
module Test_Sch( //Выходы передатчика
                 input clk,              output wire ce_tx,// Скорость (Tceo=1/Vel)
                 input [1:0]Nvel,        output wire Tx1,//Импульсы канала 1
                 input [7:0]adr,         output wire Tx0,//Импульсы канала 0
                 input [22:0]dat,        output wire SLP,// Крутизна фронтов
                 input st,               output wire en_tx_dat, // Интервал передачи данных
                                         output wire T_end_dat_tx, // Такт конца кадра
                                         output wire ft_tx, // Триггер контроля четности
                                         output wire txd, // Последовательные данные
                                         output wire qm, // Модулятор
                                         output wire [5:0]cb_bit_tx,//Счетчик бит
                                         output wire en_tx_word, //Интервал передачи слова
                       //Выходы приемника
                                         output wire[23:0] sr_dat_rx, //Регистр сдвига данных
                                         output wire[ 7:0] sr_adr_rx, //Регистр сдвига адреса
                                         output wire ok_rx ); //Успешный прием

// Модуль передатчика
AR_TXD DD1 (
                 .clk(clk),      .ce(ce_tx), // Скорость (Tceo=1/Vel)
                 .Nvel(Nvel),    .Tx1(Tx1), //Импульсы канала 1
                 .adr(adr),      .Tx0(Tx0), //Импульсы канала 0
                 .dat(dat),       .SLP(SLP), // Крутизна фронтов
                 .st(st),         .en_tx_dat(en_tx_dat), // Интервал передачи данных
                                  .T_end_dat(T_end_dat_tx), // Такт конца кадра
                                  .ft_tx(ft_tx), // Триггер контроля четности
                                  .txd(txd), // Последовательные данные
                                  .qm(qm), // Модулятор
                                  .cb_bit(cb_bit_tx),//Счетчик бит
                                  .en_tx_word(en_tx_word));//Интервал передачи слова

// Модуль приемника
AR_RXD DD2 (      .In0(Tx0),     .sr_dat(sr_dat_rx),
                  .In1(Tx1),     .sr_adr(sr_adr_rx),
                  .clk(clk),     .ce_wr(ok_rx) );
endmodule
```

4.2 Генератор последовательного включения цифр семи сегментного индикатора.

```
module Gen4an (   input clk,      output reg [1:0] q = 0,//Счетчик номера анода
                  input ce,       output wire [3:0] an );
```

```
assign an =     (q==0)? 4'b1110 ://включение цифры 0 (младшей)
                (q==1)? 4'b1101 ://включение цифры 1
                (q==2)? 4'b1011 ://включение цифры 2
                        4'b0111 ;//включение цифры 3 (старшей) 7
always @ (posedge clk) if (ce) begin
q <= q+1 ;
end
endmodule
```

4.3 Дешифратор для семи сегментного индикатора

```
module D7seg(input [3:0] dig, output wire [6:0] seg);
                        //gfedcba
assign seg =    (dig== 0)? 7'b1000000 ://    a
                (dig== 1)? 7'b1111001 :// f|    |b
                (dig== 2)? 7'b0100100 ://     g
                (dig== 3)? 7'b0110000 :// e|    |c
                (dig== 4)? 7'b0011001 ://     d
                (dig== 5)? 7'b0010010 ://
                (dig== 6)? 7'b0000010 ://
                (dig== 7)? 7'b1111000 ://
                (dig== 8)? 7'b0000000 ://
                (dig== 9)? 7'b0010000 ://
                (dig==10)? 7'b0001000 ://A
                (dig==11)? 7'b0000011 ://b
                (dig==12)? 7'b1000110 ://C
                (dig==13)? 7'b0100001 ://d
                (dig==14)? 7'b0000110 ://E
                           7'b0001110 ;//F
endmodule
```

4.4 Мультиплексор 4-х битных цифр

```
module MUX16_4 (    input [15:0] dat, output wire [3:0] do,
                    input [1:0] adr);
assign do = (adr==0)? dat[3:0]:
(adr==1)? dat[7:4]:
(adr==2)? dat[11:8]: dat[15:12];
endmodule
```

4.5 Генератор сигналов с периодом 1мс

```
module Gen1ms (input clk, //Сигнал синхронизации
                          output wire ce_1ms); //1 милисекунда
parameter Fclk =50000000 ;   //Частота генератора синхронизации 50 МГц
parameter F1kHz =1000 ;      //Частота 1 кГц
reg[16:0]ct_ms = 0 ;         //Счетчик миллисекунд
assign ce_1ms = (ct_ms==0) ;//1 милисекунда
//Делитель частоты
always @(posedge clk) begin
ct_ms <= ce_1ms? ((Fclk/F1kHz)-1) : ct_ms-1 ;//Счет миллисекунд
end
endmodule
```

4.6 Модуль генератора точки

```
module Gen_P (      input [1:0] ptr,          output wire seg_P,
                    input [1:0] adr_An );
assign seg_P = !(ptr==adr_An) ;
endmodule
```

4.7 Модуль семи сегментного индикатора

```
module DISPLAY(     input clk,                output wire [3:0] AN,
                    input [15:0]dat,         output wire [7:0] SEG,
                    input [1:0]PTR,          output wire ce1ms);
wire [3:0]Dig;
wire [1:0]Adr_dig ;
//Генератор "анодов"
Gen4an DD1( .clk(clk),        .q(Adr_dig),
            .ce(ce1ms),       .an(AN));
// Мультиплексор цифр
MUX16_4 DD2 (       .dat(dat),        .do(Dig),
                    .adr(Adr_dig));
// Дешифратор семи сегментных символов цифр
D7seg DD3 ( .dig(Dig),        .seg(SEG[6:0]));
// Генератор точки
Gen_P DD4 ( .adr_An(Adr_dig),       .seg_P(SEG[7]),
                                    .ptr(PTR) );
// Генератор ce1ms
Gen1ms DD5 (.clk(clk),        .ce1ms(ce1ms));
endmodule
```

4.8 Связь портов схемы с контактными площадками ПЛИС (файл *.ucf)

```
NET "F50MHz" LOC = "B8" ;#clk
NET "AN<0>" LOC = "F17" ;
NET "AN<1>" LOC = "H17" ;
NET "AN<2>" LOC = "C18" ;
NET "AN<3>" LOC = "F15" ;
NET "BTN0" LOC = "B18" ;
#NET "BTN1" LOC = "D18" ;
#NET "BTN2" LOC = "E18" ;
NET "BTN3" LOC = "H13" ;
NET "seg<0>" LOC = "L18" ;
NET "seg<1>" LOC = "F18" ;
NET "seg<2>" LOC = "D17" ;
NET "seg<3>" LOC = "D16" ;
NET "seg<4>" LOC = "G14" ;
NET "seg<5>" LOC = "J17" ;
NET "seg<6>" LOC = "H14" ;
NET "seg<7>" LOC = "C17" ; #DOT

NET "SW<0>" LOC = "G18" ;
NET "SW<1>" LOC = "H18" ;
NET "SW<2>" LOC = "K18" ;
NET "SW<3>" LOC = "K17" ;
```

```
NET "SW<4>" LOC = "L14" ;
NET "SW<5>" LOC = "L13" ;
NET "SW<6>" LOC = "N17" ;
#NET "SW<7>" LOC = "R17" ;
#NET "LED0" LOC = "J14" ; #LD0
#NET "LED1" LOC = "J15" ; #LD1
#NET "LED2" LOC = "K15" ; #LD2
#NET "LED3" LOC = "K14" ; #LD3
#NET "LED4" LOC = "E17" ; #LD4
#NET "LED5" LOC = "P15" ; #LD5
#NET "LED6" LOC = "F4" ; #LD6
NET "LED7" LOC = "R4" ; #LD7
#NET "TXD" LOC = "P9" ;
#NET "RXD" LOC = "U6" ;

NET "JA1" LOC = "L15" ;#Pin1
#NET "JA2" LOC = "K12" ;#Pin2
#NET "JA3" LOC = "L17" ;#Pin3
NET "JA4" LOC = "M15" ;#Pin4
NET "JA7" LOC = "K13" ;#Pin7
#NET "JA8" LOC = "L16" ;#Pin8
#NET "JA9" LOC = "M14" ;#Pin9
#NET "JA10" LOC = "M16" ;#Pin10
NET "JB1" LOC = "M13" ;#Pin1
NET "JB2" LOC = "R18" ;#Pin2
NET "JB3" LOC = "R15" ;#Pin3
NET "JB4" LOC = "T17" ;#Pin4
NET "JB7" LOC = "P17" ;#Pin7
NET "JB8" LOC = "R16" ;#Pin8
NET "JB9" LOC = "T18" ;#Pin9
#NET "JB10" LOC = "U18" ;#Pin10

NET "JC1" LOC = "G15" ;#Pin1
NET "JC2" LOC = "J16" ;#Pin2
#NET "JC3" LOC = "G13" ;#Pin3
#NET "JC4" LOC = "H16" ;#Pin4
#NET "JC7" LOC = "H15" ;#Pin7
#NET "JC8" LOC = "F14" ;#Pin8
#NET "JC9" LOC = "G16" ;#Pin9
#NET "JC10" LOC = "J12" ;#Pin10
#NET "JD1" LOC = "J13" ;#Pin1
#NET "JD2" LOC = "M18" ;#Pin2
#NET "JD3" LOC = "N18" ;#Pin3
#NET "JD4" LOC = "P18" ;#Pin4
#NET "JD7" LOC = "K14" ;#LD3
#NET "JD8" LOC = "K15" ;#LD3
#NET "JD9" LOC = "J15" ;#LD3
#NET "JD10" LOC = "J14" ;#LD3
```

5. Контрольные вопросы

5.1 Можно ли составить схему ARINC передатчика без сигнала синхронизации?
5.2 Можно ли составить схему ARINC приемника без сигнала синхронизации?
5.3 Как зависит схема ARINC приемника от скорости передачи слова?
5.4 Можно ли в ARINC приемнике измерить скорость?