ПРИМЕРЫ АППАРАТНЫХ РЕАЛИЗАЦИЙ АЛГОРИТМОВ ВЫЧИСЛЕНИЯ КОНТРОЛЬНОЙ СУММЫ CRC32

Мыцко Е.А.
Томский политехнический университет, 634050, Россия, г. Томск, пр. Ленина, 30
E-mail: evgenvt@tpu.ru, evgenrus70@mail.ru

Введение

В свободном доступе и в литературе [1] можно встретить описания программных реализации алгоритмов вычисления контрольной суммы CRC32. Однако описания аппаратных реализаций не встречаются в литературе, а только в некоторых статьях [2] без конкретных примеров. В данной статье приводятся примеры аппаратных реализаций с описанием функциональной схемы на языке VHDL.

Аппаратная реализация табличного алгоритма

На основе исходных кодов [1] программной реализации вычисления контрольной суммы CRC32 была спроектирована и описана на языке VHDL функциональная схема аппаратной реализации CRC32 в среде Quartus II с использованием программируемых логических интегральных схем (ПЛИС) Cyclone макета SDK 6.1 [3].

На рис.1 приведена функциональная схема аппаратной реализации вычисления контрольной суммы CRC32 табличным алгоритмом.

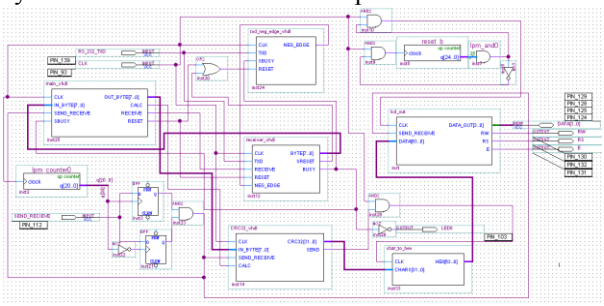

Рис. 1. Функциональная схема

Функциональная схема на рис.1 разработана с использованием блочно-ориентированного подхода [4] и состоит из 6 блоков, описанных на языке VHDL. Условно схему можно разделить на 3 части: входная часть (блоки receiver_vhdl, main_vhdl, txd_neg_edge_vhdl), блок вычисления CRC32 (CRC32_vhdl) и выходная часть (блоки char_to_hex, lcd_out).

Блок receiver_vhdl осуществляет побитовый приём данных с персонального компьютера по протоколу RS232 и формирует входной байт, который передаётся в управляющий блок main_vhdl. В момент приёма данных по последовательному порту блок txd_neg_edge_vhdl обнаруживает START_BIT входного байта.

После окончания приёма байта данных, с помощью управляющего сигнала «CALC» разрешается расчёт контрольной суммы в блоке CRC32_vhdl согласно табличному алгоритму [1]. Далее приведено описание блока CRC32_vhdl на языке VHDL.

В данном описании приведены только начальные и конечные значения таблицы, необходимой для вычисления CRC32 (table(0)…table(255)). Полное описание таблицы приведено в [5].

```
entity CRC32_vhdl is
port(
    CLK       : in std_logic;
    IN_BYTE: in std_logic_vector(7 downto 0);
    SEND_RECEIVE: in std_logic;
    CALC: in std_logic;
    CRC32: out std_logic_vector (31 downto 0);
    SEND: out std_logic );
end entity;
begin
table(0) <= X"00000000";
table(1) <= X"77073096";
……..
table(254) <= X"5A05DF1B";
table(255) <= X"2D02EF8D";
begin
if ( rising_edge(CLK)) then
    case CALC is when '1' =>
    case c_state is when s0 =>
    SEND <='0';
    index:=TO_INTEGER(unsigned(crc(7 downto 0 )
xor IN_BYTE));
    crc := crc (31 downto 8)  xor table(index);
    when others => c_state <= s0;
    end case;
    case SEND_RECEIVE is when '1' =>
    CRC32 <= not crc; SEND <= '1'; STOP := '1';
    when '0' => if ( STOP = '1' ) then
    crc:=X"FFFFFFFF"; STOP := '0';
    end if;
    end case; end if; end process; end CRC32;
```

По управляющему сигналу «SEND_RECIEVE» на выход CRC32 подаётся рассчитанная контрольная сумма. Далее контрольная сумма поступает на блок преобразования в шестнадцатеричную систему (char_to_hex), а затем на блок вывода (lcd_out) на жидкокристаллический индикатор (ЖКИ).

Аппаратная реализация матричного алгоритма

Согласно программной реализации [1] матричный алгоритм имеет несколько модификаций: однобайтовый, двухбайтовый и четырёхбайтовый. Для однобайтового матричного алгоритма функциональная схема не отличается от схемы табличного алгоритма, за исключением описания блока

вычисления CRC32 на языке VHDL, которое приведено далее.

```
matrix(0) <= X"77073096";
…….
matrix(7) <= X"EDB88320";
……
when s0 =>
    SEND <='0';
    crcb := crc( 7 downto 0 ) xor IN_BYTE;
    crc:= X"00" & crc( 31 downto 8);
    if ( crcb(0) = '1' ) then
    crc:=  crc xor matrix(0);
    …….
    if ( crcb(7) = '1' ) then
    crc:=  crc xor matrix(7);
```

В данном случае изменена матрица предвычисленных значений и основной цикл расчёта CRC. Принцип взаимодействия и синхронизации блоков функциональной схемы остаётся таким же, как и при реализации табличного алгоритма.

Для реализации двухбайтового матричного алгоритма необходимо в функциональной схеме изменить 2 блока: main_vhdl и CRC32_vhdl. Основные изменения заключаются в увеличении разрядности шины данных до 16 бит, соединяющей управляющий блок main_vhdl и блок вычисления контрольной суммы CRC32_vhdl. Далее приведено описание блока вычисления контрольной суммы CRC32 на языке VHDL. Полное описание матрицы (matrix(0)…matrix(15)) приведено в [5].

```
case SHIFT is
when '0' =>
crcb := crc( 15 downto 0 ) xor IN_BYTES;
crc:= X"0000" & crc( 31 downto 16 );
when '1' =>
crcb:= crc( 15 downto 0 ) xor IN_BYTES;
crcb := crcb ( 7 downto 0 ) & X"00";
crc:= X"00" & crc (31 downto 8);
if ( crcb(0) = '1' ) then
crc:=  crc xor matrix(0);
……
if ( crcb(15) = '1' ) then
crc:=  crc xor matrix(15);
```

Для аппаратной реализации четырёхбайтового матричного алгоритма необходимо по аналогии с двухбайтовым матричным алгоритмом заменить управляющий блок и блок вычисления CRC32 в связи с увеличением разрядности шины до 32 бит, а также изменить описание блока CRC32_vhdl согласно алгоритму. Полное описание матрицы (matrix(0)…matrix(31)) приведено в [5].

```
case SHIFT is
when "01" =>
crcb:= crc xor IN_BYTES ( 7 downto 0 );
crcb := crcb ( 7 downto 0 ) & X"000000";
crc:= X"00" & crc (31 downto 8);
when "10" =>
crcb:= crc xor IN_BYTES ( 15 downto 0 );
crcb := crcb ( 15 downto 0) & X"0000";
crc:= X"0000" & crc (31 downto 16);
when "11" =>
crcb:= crc xor IN_BYTES ( 23 downto 0 );
crcb  := crcb ( 23 downto 0) & X"00";
crc:= X"000000" & crc (31 downto 24);
end case;
if ( crcb(1) = '1' ) then
crc:=  crc xor matrix(1);
……
if ( crcb(31) = '1' ) then
crc:=  crc xor matrix(31);
```

Таким образом, используя приведённые примеры аппаратной реализации, можно создавать аппаратуры для вычисления контрольной суммы CRC32.

Заключение

В данной статье приведены примеры аппаратных реализаций вычисления контрольной суммы CRC32 с применением ПЛИС Cyclone макета SDK 6.1 для табличного, матричного, двухбайтового и четырёхбайтового матричного алгоритмов. При проектировании функциональной схемы использовался блочно-ориентированный подход (BBD). Блоки функциональной схемы receiver_vhdl, main_vhdl, txd_neg_edge_vhdl, CRC32_vhdl, char_to_hex, lcd_out разработаны с применением языка описания аппаратуры VHDL. Показана практическая значимость приведённых примеров аппаратной реализации для применения их на ПЛИС Cyclone от Altera.

Литература

1. Мыцко Е.А., Мальчуков А.Н. Исследование программных реализаций табличного и матричного алгоритмов вычисления контрольной суммы CRC32 [Электронный ресурс] // Вестник науки Сибири. Серия: Информационные технологии и системы управления. – 2011 – №. 1 – С. 273-278. – URL: http://sjs.tpu.ru/journal/issue/view/2/showToc/sect/4 (дата обращения 02.10.14).

2. Особенности аппаратной реализации алгоритмов вычисления контрольной суммы CRC32 [Электронный ресурс] // Вестник науки Сибири. - 2012 - №. 5 (6) - С. 87-92. - Режим доступа: http://sjs.tpu.ru/journal/article/view/513.

3. Учебный лабораторный стенд SDK-6.1 // Embedded systems. [Электронный ресурс]. URL: http://embedded.ifmo.ru/index.php/support/sdk-61 (дата обращения: 02.10.2014).

4. О применении блочно-ориентированного подхода к разработке устройств на ПЛИС [Электронный ресурс] // Вестник науки Сибири. Серия: Информационные технологии и системы управления. - 2011 - №. 1 - С. 379-381. - Режим доступа:http://sjs.tpu.ru/journal/issue/view/2/showToc/sect/4.

5. Мыцко Е. А., Мальчуков А. Н., Осокин А. Н. Контрольная сумма CRC. Исследование алгоритмов вычисления контрольной суммы CRC. - Saarbrucken: LAP LAMBERT Academic Publishing GmbH & Co. KG, 2013 - 180 с.