International Journal of Engineering Research and Development
***e**-ISSN: 2278-067X, **p**-ISSN: 2278-800X, www.ijerd.com*
Volume 4, Issue 5 (October 2012), PP. 13-18

Synchronization of Time Code Format Generator to Computer with RS-232 Using UART Developed in VHDL

DasariKiran[1], A.V.V.Prasad[2], D.Srikar[3]
[1]M.Tech, BVRIT, JNTU , Medak, Andhrapradesh, India.
[2]Head, OBSSR, DPO&AD/NRSC, Hyderabad, India-500018.
[3] Assistant Professor BVRIT, JNTU , Medak, Andhrapradesh, India

Abstract:—National Remote Sensing Center (NRSC) receives data from different remote satellites like IRS-P6, IRS-P5, Cartosat-2, Cartosat-2a, etc., and processes it depending on the user requirements. The satellite data received in X band is in a particular data format. This data has to be frame synchronized using a special hardware. This hardware needs time information in a special format. This time information is added in every line by the frame synchronization hardware. In the proposed project VHDL code has been developed for the generation of time in days, hours, minutes, seconds, milliseconds, microseconds structure in a BCD format. Computer will provide the start time. This time will be interfaced to the developed hardware using the UART developed within the ALTERA EPLD. The time increments will be displayed on HP display devices. The developed hardware will continuously increment from the start time provided by the computer at an interval of 1micro second. This project has been implemented and tested using the ALTERA EPLDs. This needs RS232 settings like baud rate, parity, start bits, stop bits, etc., must be programmed as per the requirement. The hardware required for this has been implemented on the wire-warp board.

Keywords:—Very High Speed Integrated Circuit Hardware Description Language (VHDL), Binary Coded Decimal (BCD), Universal Asynchronous Receiver And Transmitter (UART), Electrically Programmable Logic Device (EPLD)

I. INTRODUCTION

The IRIG-B encoder and its associated modulator perform time code generation. The encoder consists of a two level multiplexer and a series of ancillary gates. These elements employ the contents of minor and major time counters to formulate the code format and to insert the appropriate time information. The encoder produces a dc-level shift version of code, which is applied to both the modulator and an output connector. The modulator uses a 1KHz sine wave to modulate the code signal, which is also routed to an output connector.

The major and minor counters accumulate the basic timing information. The minor chain is clocked by crystal-controlled oscillator and accumulates hundredths-of-the millisecond through tens-of-seconds. The major time counter accumulates unit-of-seconds through hundredth-of-days. In addition the major and minor time counters are loaded with an initial value when the operator presets the start time before initiating code generation.

For precise processing and framing of the satellite data, an accurate time reference source is needed. The time reference is generated by equipment called Time Code Generator (TCG). It generates time of the year DDD: HH: MM: SS: MsMsMs: McMcMc format. It outputs the generated time both parallel and serial forms. The serial output of the TCG is distributed to various time code translators, which converts the serial time back to parallel form.

II. EXISTING TECHNIQUE

From GPS also time can be acquired and tag to the satellite data. The problem with the GPS satellite is time information will be in DDD: HH: MM: SS format that mean that the received time is up to seconds only which has less time resolution. As NRSC satellites receive data for every 0.35 microsecond. These hardware components require time information in microseconds also.

III. SCOPE OF THE PROJECT

Figure 1 shows the block diagram of a satellite data acquisition system. The QPSK modulated PCM serial stream is captured by receiving antenna system, the received data is down converted into an immediate frequency while preserving the base band data is recovered after demodulating the IF signal. An NRZ-L (non return to zero level) data and clock are recovered through the bit synchronization. The data and clock are fed to the Front End Hardware (FEH). The FEH basically detects the start of each satellite data frame and converts the serial data into parallel form and transfers over the system bus and the data is stored onto a secondary storage.

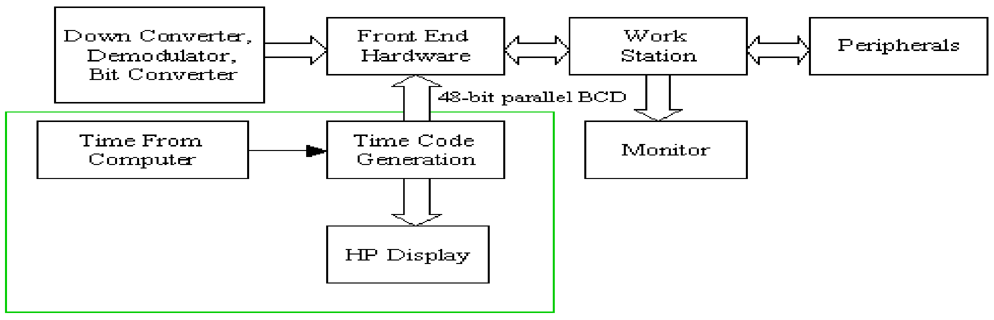


Fig. 1 Block diagram of time code generator

TCG receives time from the computer through UART (Universal Asynchronous Receiver And Transmitter) and counts from that reference time. It outputs the generated time both parallel and serial forms. The serial output of the TCG is distributed to various time code translators, which converts the serial time back to parallel form. The parallel BCD time output from the TCT (Time Code Translator) is connected to the input of the front-end hardware. The FEH tags each satellite scan line/frame with the time available from the TCT at that instant. Hence it becomes necessary to generate an accurate time with a resolution at least 0.1ms. In future it varies up to 0.01ms.

IV. UART

Asynchronous transmission allows data to be transmitted without the sender having to send a clock signal to the receiver. Instead, the sender and receiver must agree on timing parameters in advance and special bits are added to each word that is used to synchronize the sending and receiving units.

When a word is given to the UART for Asynchronous transmissions, a bit called the "Start Bit" is added to the beginning of each word that is to be transmitted. The Start Bit is used to alert the receiver that a word of data is about to be sent, and to force the clock in the receiver into synchronization with the clock in the transmitter. These two clocks must be accurate enough to not have the frequency drift by more than 10% during the transmission of the remaining bits in the word. After the Start Bit, the individual bits of the word of data are sent, with the Least Significant Bit (LSB) being sent first. UART frame format is as shown in figure 2. Each bit in the transmission is transmitted for exactly the same amount of time as all of the other bits, and the receiver "looks" at the wire at approximately halfway through the period assigned to each bit to determine if the bit is a 1 or a 0. For example, if it takes two seconds to send each bit, the receiver will examine the signal to determine if it is a 1 or a 0 after one second has passed, then it will wait two seconds and then examine the value of the next bit, and so on. The sender does not know when the receiver has "looked" at the value of the bit. The sender only knows when the clock says to begin transmitting the next bit of the word.

Start	Data 0	Data 1	Data 2	Data 3	Data 4	Data 5	Data 6	Data 7	Stop

Fig2: UART Frame Format

When the entire data word has been sent, the transmitter may add a Parity Bit that the transmitter generates. The receiver to perform simple error checking may use the Parity Bit. The transmitter then sends at least one Stop Bit. When the receiver has received all of the bits in the data word, it may check for the Parity Bits (both sender and receiver must agree on whether a Parity Bit is to be used), and then the receiver looks for a Stop Bit. If the Stop Bit does not appear when it is supposed to, the UART considers the entire word to be garbled and will report a Framing Error to the host processor when the data word is read. The usual cause of a Framing Error is that the sender and receiver clocks were not running at the same speed, or that the signal was interrupted. Regardless of whether the data was received correctly or not, the UART automatically discards the Start, Parity and Stop bits. If the sender and receiver are configured identically, these bits are not passed to the host. If another word is ready for transmission, the Start Bit for the new word can be sent as soon as the Stop Bit for the previous word has been sent. Because asynchronous data will be "self synchronizes", if there is no data to transmit, the transmission line can be idle. The computer provides the reference time to the TCG through UART. The functional block diagram of UART is as shown in figure3.

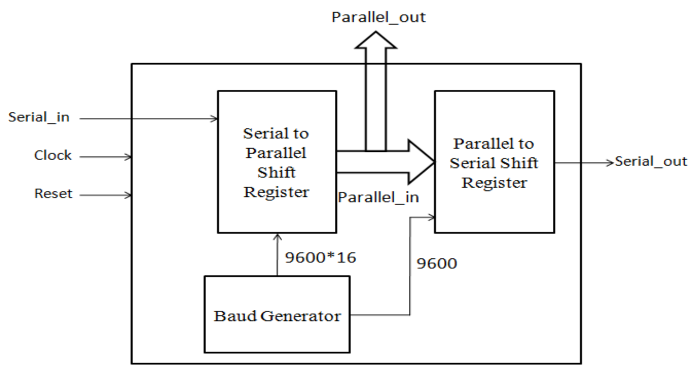


Fig3: UART Block Diagram

4.1 UART Transmitter

Transmission operation is simpler since it is under the control of the transmitting system. As soon as data is deposited in the shift register after completion of the previous character, the UART hardware generates a start bit, shifts the required number of data bits out to the line, generates and appends the parity bit (if used), and appends the stop bits. Since transmission of a single character may take a long time relative to CPU speeds, the UART will maintain a flag showing busy status so that the host system does not deposit a new character for transmission until the previous one has been completed; this may also be done with an interrupt. Since full-duplex operation requires characters to be sent and received at the same time, practical UARTs use two different shift registers for transmitted characters and received characters.

4.2 UART Transmitter

All operations of the UART hardware are controlled by a clock signal, which runs at a multiple (say, 16) of the data rate - each data bit is as long as 16 clock pulses (figure 4). The receiver tests the state of the incoming signal on each clock pulse, looking for the beginning of the start bit. If the apparent start bit lasts at least one-half of the bit time, it is valid and signals the start of a new character. If not, the spurious pulse is ignored. After waiting a further bit time, the state of the line is again sampled and the resulting level clocked into a shift register. After the required number of bit periods for the character length (5 to 8 bits, typically) has elapsed, the contents of the shift register are made available (in parallel fashion) to the receiving system.

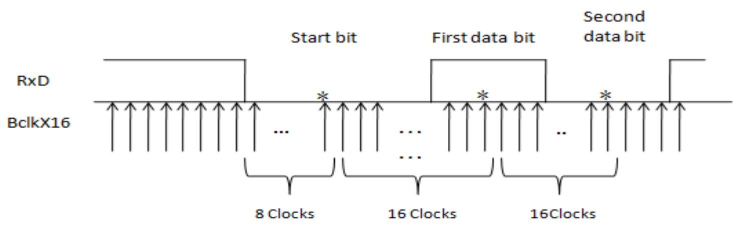


Fig4: UART Block Diagram

The UART will set a flag indicating new data is available, and may also generate a processor interrupt to request that the host processor transfers the received data. In some common types of UART, a small first-in, first-out FIFO buffer memory is inserted between the receiver shift register and the host system interface. This allows the host processor more time to handle an interrupt from the UART and prevents loss of received data at high rates.

V. PARALLEL BCD TIME OF THE YEAR GENERATION

Figure 5 shows block diagram of parallel BCD Time Code Generator. This block uses an external 10MHz crystal controlled oscillator and generates one pulse per second (1 PPS) signal. The 1 PPS signal is further used in deriving the time of the year with 1 sec resolution. Parallel BCD time generation code consists of two counters chains namely Major Chain and Minor Chain. The Minor Chain derives input from 10MHz crystal controlled oscillator. The 10MHz signal is divided by a series of decade counters connected in cascade (7 numbers) to derive a signal. The first stage is generation of microseconds, second stage is generation of millisecond; third stage is generation of second (1sec).

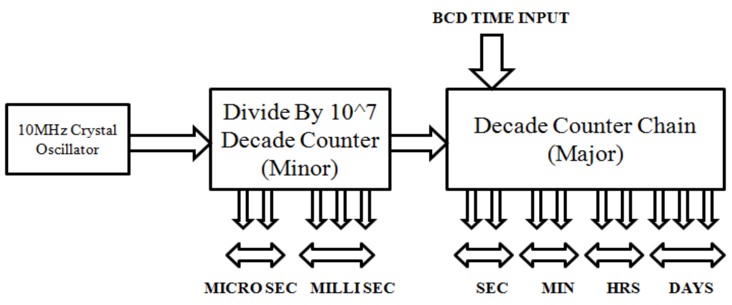


Fig. 5 Block diagram of parallel BCD Time Code Generator

The Major Chain consists of nine-decade counters connected in cascade. The first counter is simple decade counter that has to count units of sec. The output of second counter is mod 6 counter that counts from 0-5 it generate the output tens of second. The third counter is simple decade counter that generates units of min. The fourth counter is mod 6 counter that counts from 0-5 it generate the output tens of min. The fifth counter is decade counter who's output is a unit of hours. In the same way the other counters perform the same operation. The hour's counters are configured in such a way that when the time is 23:59:59, the units and tens are reset to zero as desired. Counter numbers 7, 8, 9 are configured to count from 1 to 365. The counting is enabled whenever the previous six counters overflow to all zeros.

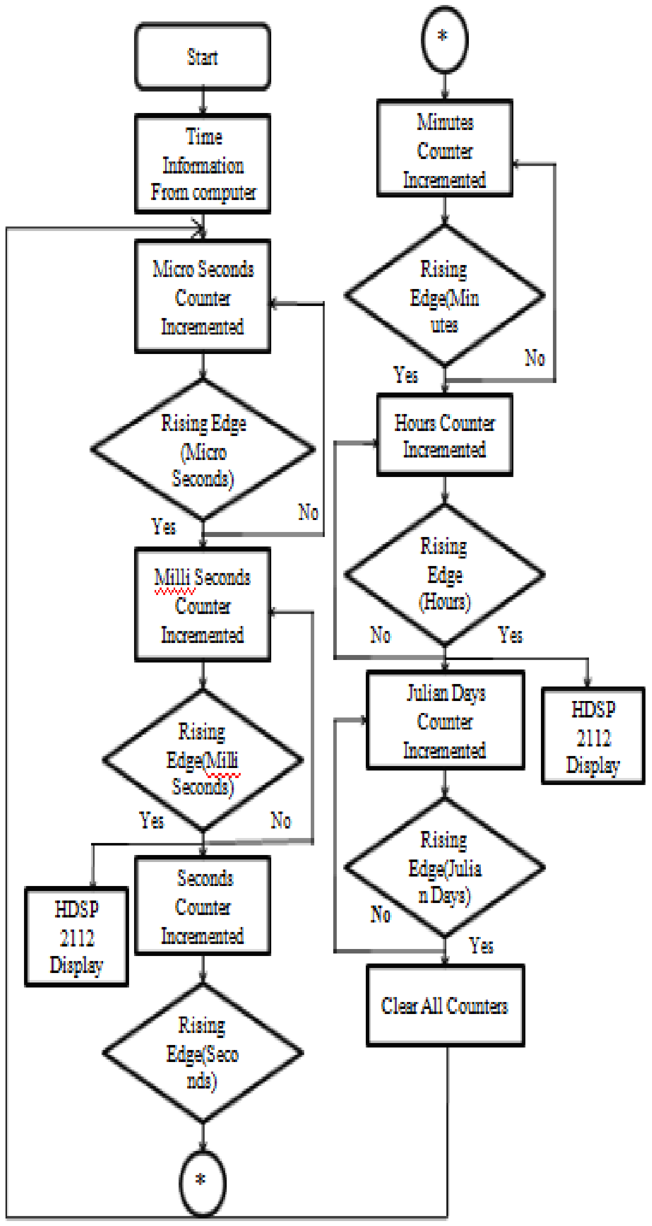
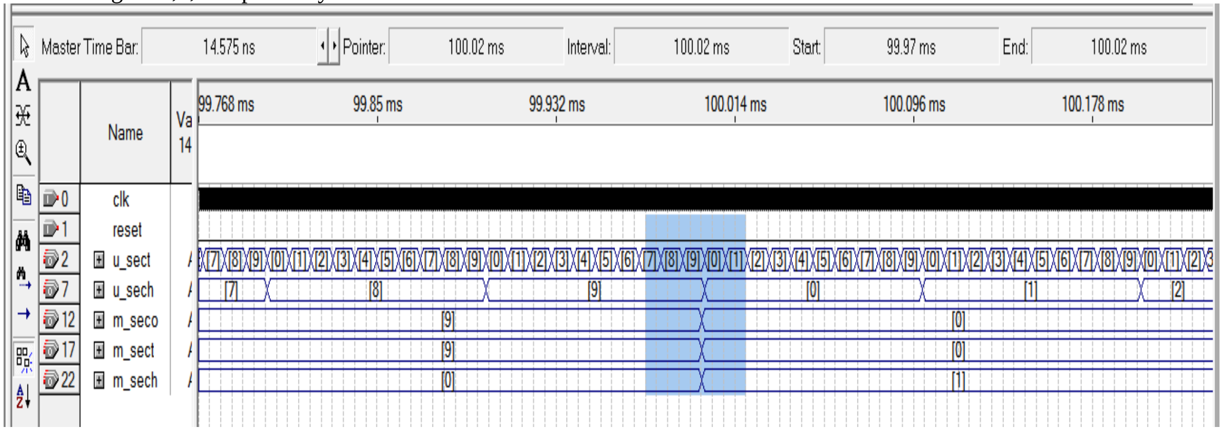

Start
Time Information From computer
Micro Seconds Counter Incremented
Rising Edge (Micro Seconds)
Yes
No
Milli Seconds Counter Incremented
Rising Edge(Milli Seconds)
Yes
No
HDSP 2112 Display
Seconds Counter Incremented
Rising Edge(Seconds)
*
*
Minutes Counter Incremented
Rising Edge(Minutes)
Yes
No
Hours Counter Incremented
Rising Edge (Hours)
No
Yes
Julian Days Counter Incremented
HDSP 2112 Display
Rising Edge(Julian Days)
No
Yes
Clear All Counters

Fig6: System Flow Diagram

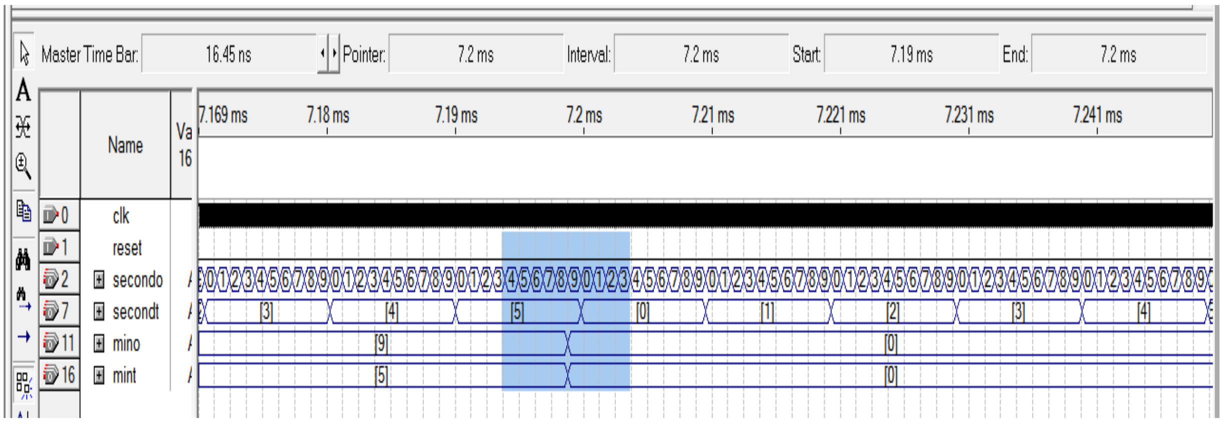
VI. RESULTS

The simulation output for TCG, micro-seconds to milli-seconds, seconds to minutes and hours to julian days are as shown in figure 7,8,9 respectively.



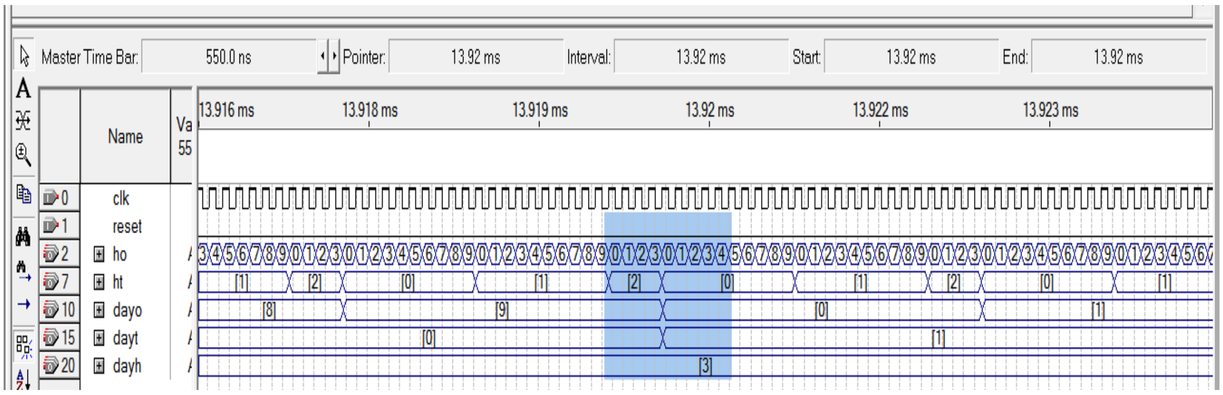
Master Time Bar: 14.575 ns Pointer: 100.02 ms Interval: 100.02 ms Start: 99.97 ms End: 100.02 ms

Fig7: Micro-Seconds to Milli-Seconds



Master Time Bar: 16.45 ns Pointer: 7.2 ms Interval: 7.2 ms Start: 7.19 ms End: 7.2 ms

Fig8: Seconds to Minutes



Master Time Bar: 550.0 ns Pointer: 13.92 ms Interval: 13.92 ms Start: 13.92 ms End: 13.92 ms

Fig. 9 Hours to Julian days

VII. CONCLUSIONS

The time code format starting from micro seconds to Julian day (micro-seconds, milli-seconds, seconds, minutes, hours and Julian day) were implemented in the ALTERA EPLD EPM7160SLC84-7 and the HDSP 2112 display was interfaced with it. The code for the time code generation, UART and the address generation were written in VHDL. The simulation was carried out with the basic clock of 10 MHz. the simulation results are tested at different phases i.e., at micro-seconds, milli-seconds, seconds, minutes, hours and Julian day.

The EPLD is fused using the standalone programmer, testing was carried out on wire wrap board. The output results were captured using TLA74 logic analyzer and compared with simulation results and found that they are matched with expected simulation. The seconds, minute’s hours and Julian days are displayed on the HDSP 2112 display. The outputs are provided in BCD format on a 64-pin flat connector, which can be used to ping time information to the data received from the satellites. The input for setting the time is taken from computer the outputs were checked for different conditions.

ACKNOWLEDGMENT

I wish to avail this opportunity to express my sincere gratitude to Dr. V. K. Dadhwal, Director, and NRSC for giving me this opportunity to do the project at NRSC.

REFERENCES

[1]. Charles H. Roth, Jr, Digital System Design by using VHDL, PWS Publishing Company, 1998.
[2]. .Wakerly, John F, Digital Design (Principles and Practices), Third Edition, Prentice-Hall United States of America, 2000.
[3]. .Altera Max PlusII version 7.1. Dec 1996.
[4]. .J. Norhuzaimin and H.H Maimun “The Design of High Speed UART” Asia-Pacific Conference on Applied Electromagnetic (APACE 2005). Dec. 2005.
[5]. .Himanshu Patel, Sanjay Trivedi, R. Neelkanthan, V. R. Gujraty “A Robust UART Architecture Based on Recursive Running Sum Filter for Better Noise Performance” International Conference on VLSI Design (VLSID'07).
[6]. .http://www.rentron.com/Files/hdsp-2112.pdf.