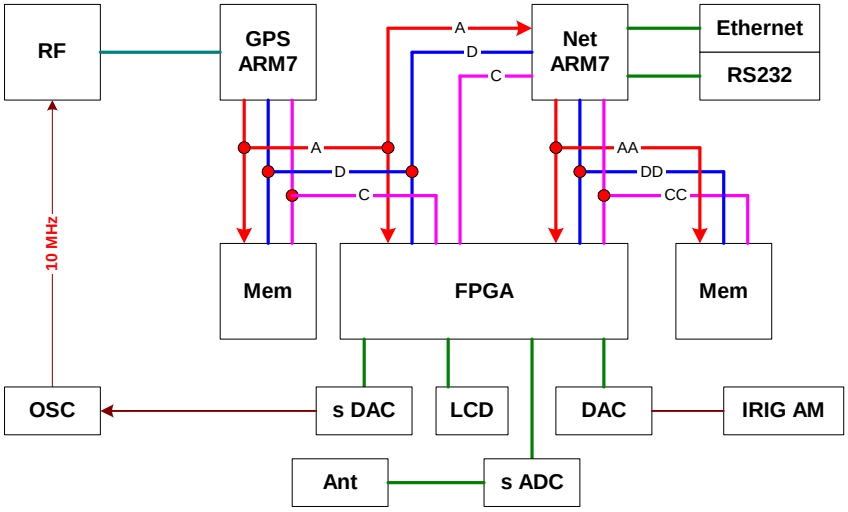
Notes on Columbia Design

©Duy-Ky Nguyen **2002-May-01**

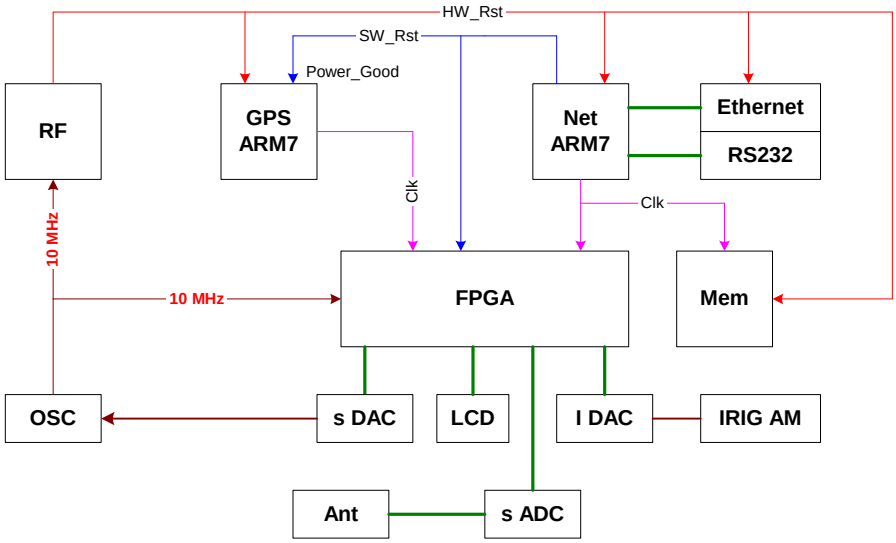
Table of Contents

1. COLUMBIA PROJECT

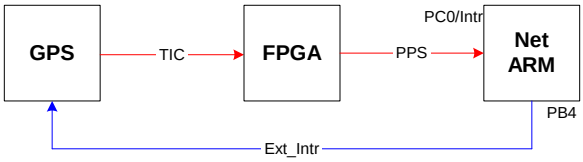
1.1. Overview



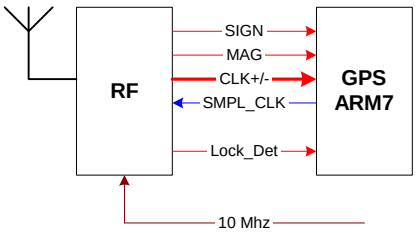
1.2. Reset & Clock



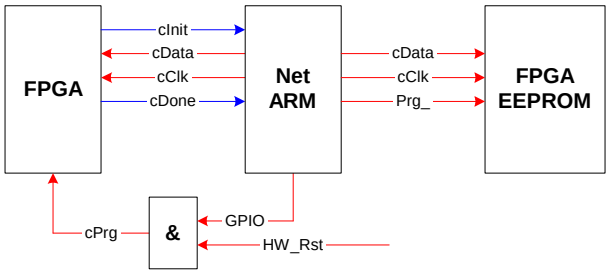
1.3. Interrupt



1.4. RF Front End

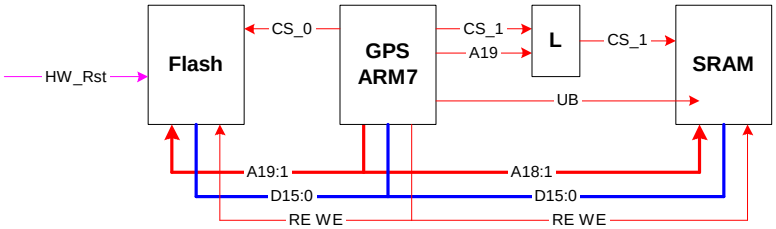


1.5. FPGA Config

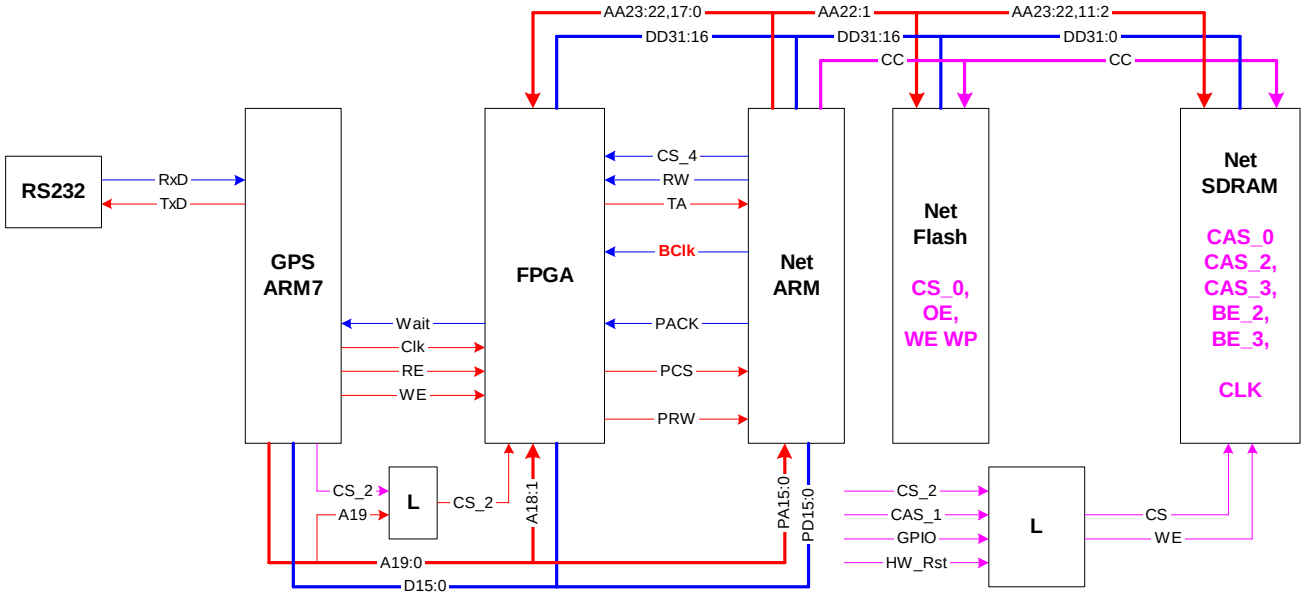


Net ARM is used to program both FPGA and its EEPROM

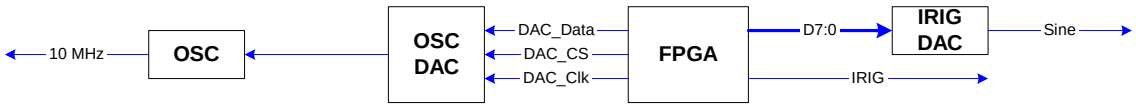
1.6. GPS Memory



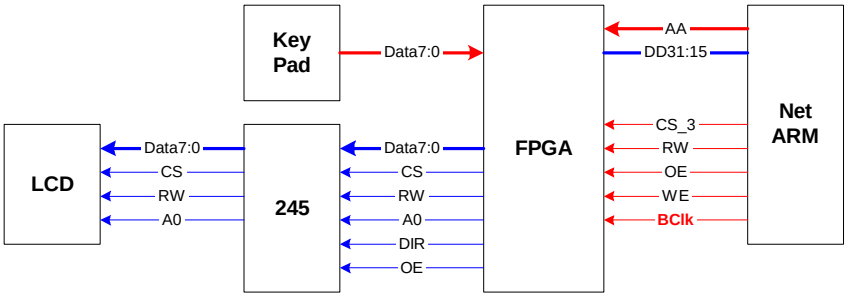
1.7. Net Memory



1.8. Oscillator & IRIG



1.9. User Interface: Key-Pad & LCD



2. MEMORY MAP

2.1. GPS ARM

Address	Size	Type	
0x0000 0000 ~ 0x0007 FFFF	512 KB	16-bit	**CS0**, Flash
0x0008 0000 ~ 0x1FFF FFFF	N.U.		
0x2000 0000 ~ 0x2007 FFFF	512 KB	16-bit	**CS1**, SRAM
0x2008 0000 ~ 0x3FFF FFFF	N.U.		
0x4000 0000 ~ 0x4007 FFFF	N.U.		CS2, Reserved (12-Channel Correlator, Peripheral Control Logic, Real Time Clock, System Clock Generator, 1PPS Timemark Generator reflected every 0x2000)
0x4008 0000 ~ 0x400F FFFF		16-bit	**CS2**, FPGA External Chip Select 2A via NSCS[2A], gated with SADD[19] = "1"
0x4010 0000 ~ 0x4010 0FFF			**CS2**, 12-Channel Correlator
0x4010 1000 - 0x4010 1FFF			**CS2**, Peripheral Control Logic, Real Time Clock, System Clock Generator,1PPS Timemark Generator
0x4010 2000 - 0x5FFF FFFF			*CS2, Reserved (CS2, Correlator, Peripherals reflected)*
0x6000 0000 - 0x6000 1FFF		32-bit	Internal SRAM - 2k x 32-bit
0x6000 2000 - 0x7FFF FFFF			*Not Used (CS3 reflected)*
0x8000 0000 - 0xDFFF FFFF			*Reserved*
0xE000 0000 - 0xE00F FFFF			Firefly MF1 bus modules
0xE010 0000 - 0xFFFF FFFF			*Reserved*

2.2. Net ARM

Chip Select	Mem Size	Data Size	
CS0	8 MB	D[31:16]	22-bit Address, 16-bit Data Flash D[31:15]
CS1		16-bit	SRAM, **NU**
CS2	16 MB	D[31:0]	SDRAM
CS3		D[7:0]	User Interface Key-Pad, LCD
CS4		D[31:16]	FPGA

2.3. FPGA Memory Map

Address Offset	**Size**	**Type**	**Description**
0x0 0000		16-bit	Internal Regsiters
0x0 0400	512 B (9-bit)	16-bit	Sine ROM Table (Addr[10] = 1)
0x0 1000	2 KB (16-bit)	16-bit	FPGA Internal Dual-Port RAM
0x1 0000	64 KB (16-bit)	16-bit	Shared Memory of Net ARM (Addr[16] = 1)

2.4. FPGA Registers

Address	Type	Description
		Basic
0x0 0000*2	RO	Chip ID (Read Only)
0x0 0001*2	RW	Control Reg (1) Read to clear 15:8 = Count ***N*** for Advance or Retard 7:3 = NU 1 = Load Counter 1 = Retard ***N*** count 0 = Advance ***N*** count
0x0 0002*2	RW	Low Counter Reg 16-bit LSB of 18-bit Counter
0x0 0003*2	RW	Index and High Counter reg 15~13 = 2-bit index 12~2 = N.U. 1~0 = 2-bit MSB of 18-bit Counter
0x0 0004*2	RO	Low Latched Counter Reg 16-bit LSB of 18-bit Counter
0x0 0005*2	RO	High Latched Counter reg 15~13 = 2-bit index 12~2 = N.U. 1~0 = 2-bit MSB of 18-bit Counter
(0x0 0005 ~ 0x0 000F)*2		*Reserved*
		IRIG
0x0 0010*2		Sec-Min IRIG Reg 15 = NU 14~8 = Second (0~59) 7 = NU 6~0 = Minute (0~59)
0x0 0011*2		Hour-DayLo IRIG Reg 15~14 = NU 13~8 = Hour (0~23) 7~0 = Control
0x0 0012*2		Day 15~9 = NU 8~0 = Day (0~365)
(0x0 0011 ~ 0x0 0017)*2		*Reserved*
0x0018*2	RO	Instant AM Sine Look-Up Table Info 15~8 = Access Address 7~0 = Data Read
(0x0 0019 ~ 0x0 001F)*2		*Reserved*
		SPI
0x0020*2	RW	ADC for Antenna (2) 15~8 = Wr: Command Byte to start ADC requires 16+ SPI_clk, Rd: ADC result requires 3+ Clk10M 7~0 = NU
0x0021*2	W	DAC for Xtal. No read-back feature provided from this device. 15~0 = Data Write
(0x0 0022 ~ 0x0 01FF)*2		*Reserved*
		IRIG Sine Look-Up Table
0x0 0400 ~ 0x4FF	RO	250-byte ROM for Low-Amp Sine Look-Up Table. Read 125 16-bit data.
0x0 0500 ~ 0x5FF	RO	250-byte ROM for High-Amp Sine Look-Up Table. Read 125 16-bit data.
0x0 0600 ~ 0x0 0FFF		*Reserved*
		DP RAM
0x0 1000 ~ 0x0 11FF	GPS ARM RW Net ARM RO	512 16-bit DP RAM
0x0 1200 ~ 0x0 1FFF		*Reserved*
0x0 2000 ~ 0x21FF	Net ARM RW GPS ARM RO	512 16-bit DP RAM

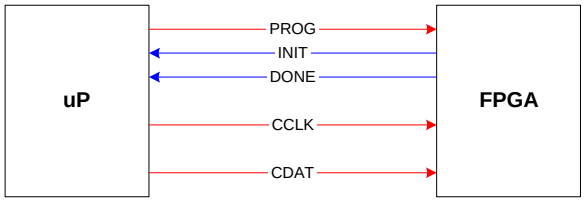
Note

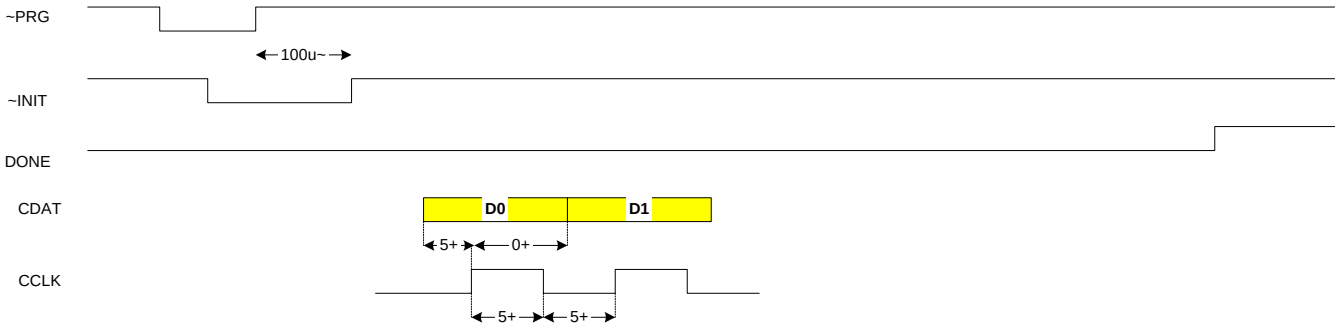
(1) Value is used once for each write, so it's clear within 5 us (counter clock), enough time to read back for verification, if so required;

(2) 8 clocks are required for 8-bit SPI command to start ADC, another 8 clock for conversion, at least. So after at least 16 clocks, a normal read can get the result;

3. CONFIGURING XILINX FPGA USING UP

3.1. Slave Serial





Notes:

"+" sign means "plus" ☺, eg "5+" means 5 is minimum

"-" sign means "minus" ☺, eg "5-" means 5 is maximum

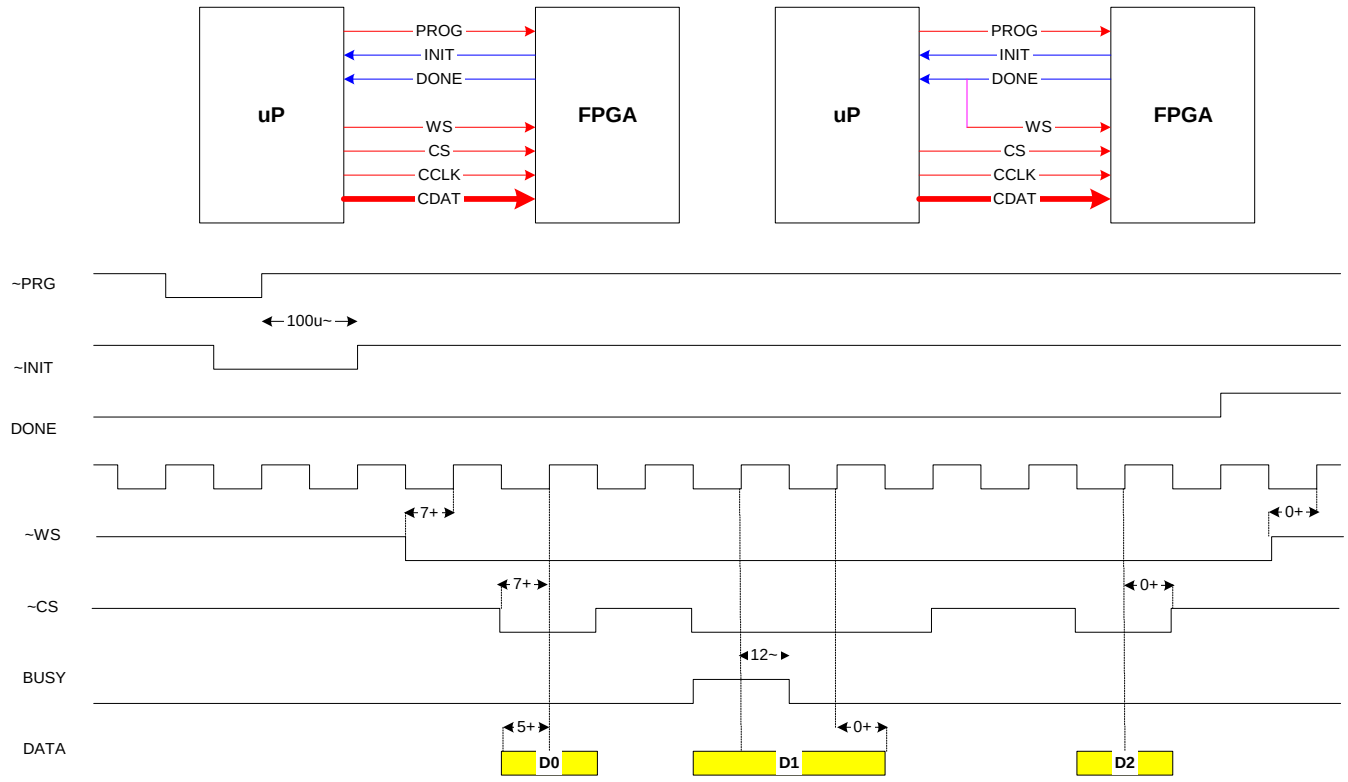
3.1.1 Requirements

- All 3 signals PROG, INIT, DONE must be pulled high
- All 5 signals are connected to general GPIO of uP
- *FPGA image: BIT file must be used*

3.1.2 Implementation

1. uP drives PROG low to start configuration
2. uP drives PROG high as required
3. uP checks INIT must be high as FPGA is ready for configuration
4. uP sends out CDAT, MSB first
5. uP drives CCLK high to tell FPGA data are valid
6. uP drives CCLK low as required
7. Repeat (4)~(6) until DONE going high for MAX_CNT in case of problem

3.2. Slave Parallel



Data are accepted @ 1st posedge clk when BUSY low

3.2.1 Requirements

- All 5 PROG, INIT, DONE, WS, CS must be pulled high (Busy is unnecessary under 50 MHz, but it's not working for now)
- CS must be CS of uP to tell FPGA data are valid, it must have **32-bit, 0 wait-state and wr_sync**
- CCLK should be system clock (33 MHz from Net ARM)
- All 5 signals are connected to general GPIO of uP
- *FPGA image: EXO file must be used* (S2record) where D0 is MSB, not LSB. It seems to me that little-endian has both bit-order and byte-order different than big-endian. We use uP little-endian, but configuration job, like other communication jobs, uses big-endian;

3.2.2 Implementation

1) uP drives PROG low to start configuration
2) uP drives PROG high as required
1) uP checks INIT must be high as FPGA is ready for configuration
2) uP drives WS low to start
5) uP sends out 8-bit CDAT via data bus until DONE going high for MAX_CNT in case of problem
3) uP drives WS high to terminate

If Done is used for WS

1) uP drives PROG low to start configuration, Done is low used for WS to start
2) uP drives PROG high as required

3) uP checks INIT must be high as FPGA is ready for configuration
4) uP sends out 8-bit CDAT via data bus until DONE going high for MAX_CNT in case of problem. Done is high used for WS to stop

4. IRIG-B PROTOCOL

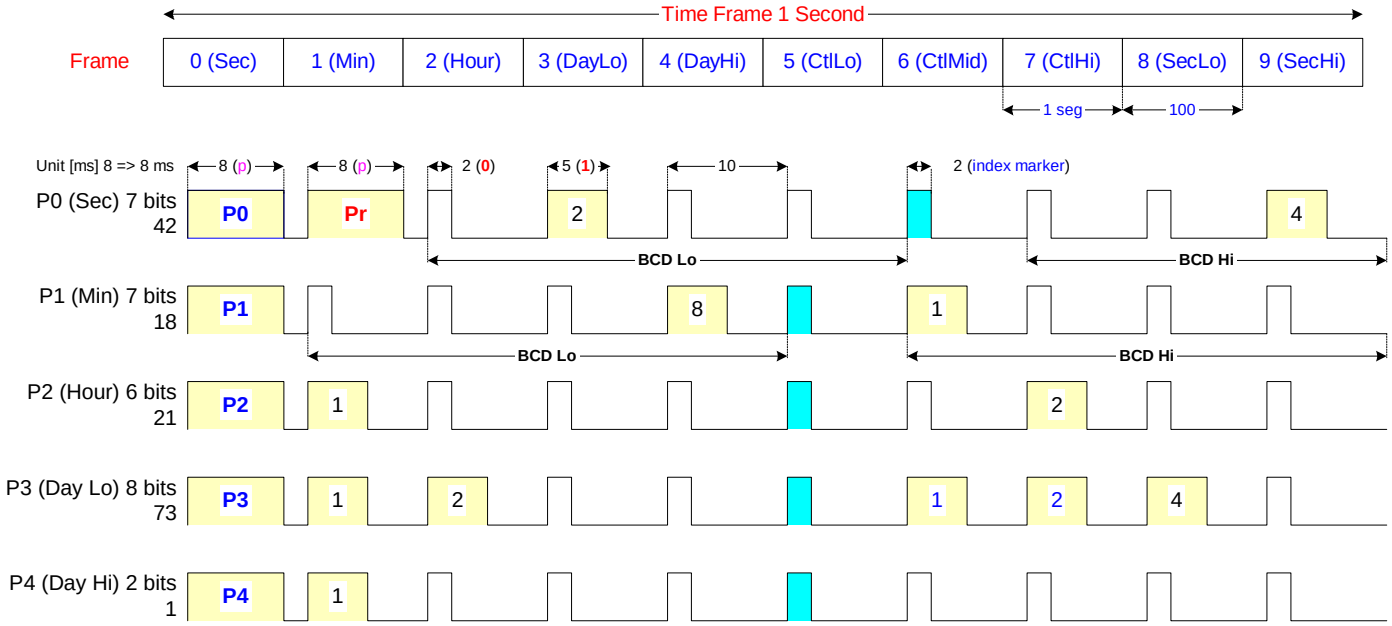
1 second (1000 ms) has one frame. One frame has 10 segments 100 ms each. One segment has 10 sub-segments 10 ms each.
Each segment first has

- 1 position marker (P0 ~ P9) of 8 ms, but the first segment has an additional reference marker also of 8 ms after P0 to mark start of frame
- 2 BCD numbers at most separated by index marker of 2 ms with LSB first. Value 1 is of 5 ms and value 0 of 2 ms.

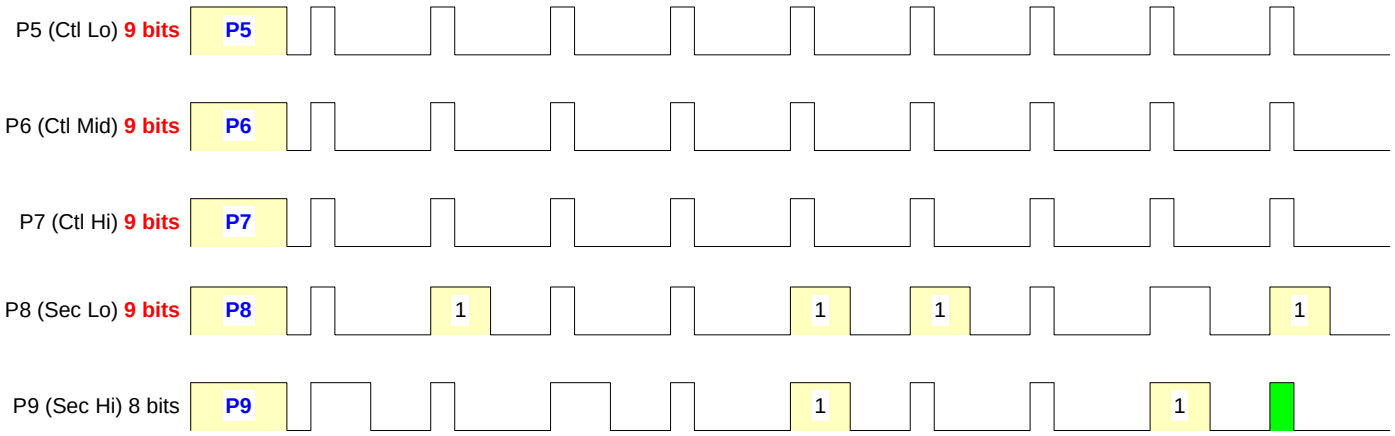
Each frame has 74 bits

- 30-bit yearly time from P0 to P4 where 366 days presented in Day_Lo and Day_Hi.
  - Seconds segment of 7 bits: 0 ~ 59: so the BCD_Lo is of 4 bit (0~9) and BCD_Hi of 3 bits (0~5)
  - Minute segment of 7 bits: 0 ~ 59: so the BCD_Lo is of 4 bit (0~9) and BCD_Hi of 3 bits (0~5)
  - Hour segment of 6 bits: 0 ~ 23: so the BCD_Lo is of 4 bit (0~9) and BCD_Hi of 2 bits (0~2)
  - Day_Lo segment of 8 bits: 0 ~ 99: so the BCD_Lo and BCD_Hi are of 4 bit (0~9)
  - Day_Hi segment of 2 bits: 0 ~ 3: so the BCD_Lo and BCD_Hi are of 2 bit (0~3)

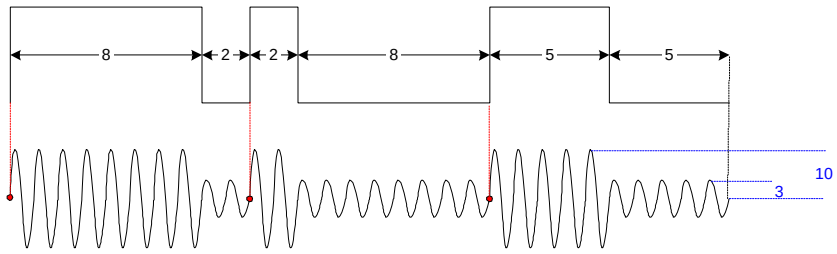
- 27-bit control: P5 ~ P7, 9 bits each segment. No index marker
- 17-bit daily seconds as Straight Binary Seconds (SBS): 24*3600 = 86400 = $15180 (17-bit wide). No index marker.

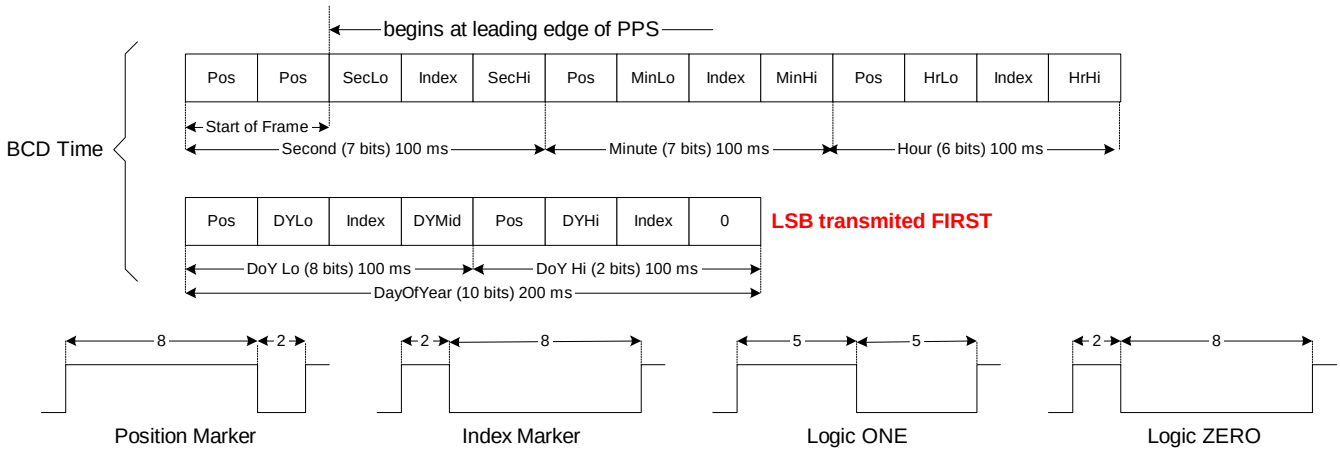


So we have 173 days, 21 hours, 18 minutes, 42 seconds.



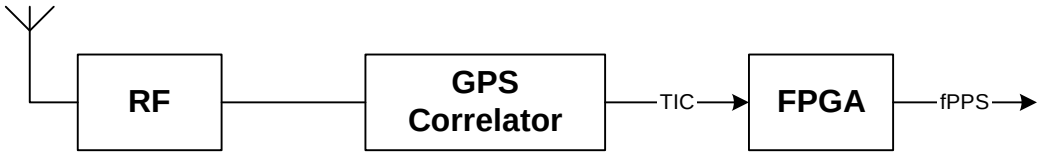
Within the day above, we have (21 hr + 18 min + 42 sec) = 76722 = $12bb2 presented in P8 (9 bits) & P9 (8 bits) with LSB first.
The pulse-width signal above will be *amplitude-modulated* the mark-space ration 10:3.
The positive-going zero-crossing sine wave must be coincident with positive edge of the pulse.





5. NOTES ON STEERING TIME COUNTER FOR PPS SIGNAL

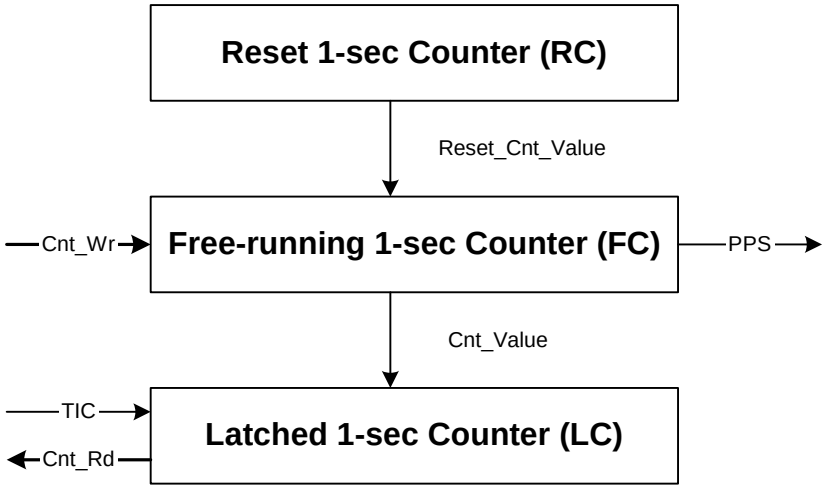
5.1. Problem Statement



GPS Correlator and ARM7 processor are combined in GP4020 from Zarlink. TIC is a PPS generated by GPS Correlator when locked to satellite. So, PPS must be free-running PPS when no TIC (no satellite locked) and synchronized to it when available.

5.2. Algorithm

This algorithm is a simpler version implemented in NTS-200 where GPS processor is not required to compute mSec based on time from SV(Space Vehicle, satellite)



Assumption
Assume, first, free-running counter FC counts 100 from 10 to (10+99) to make 1 second clocked by 100 Hz. So, Reset_Cnt_Value is 10.

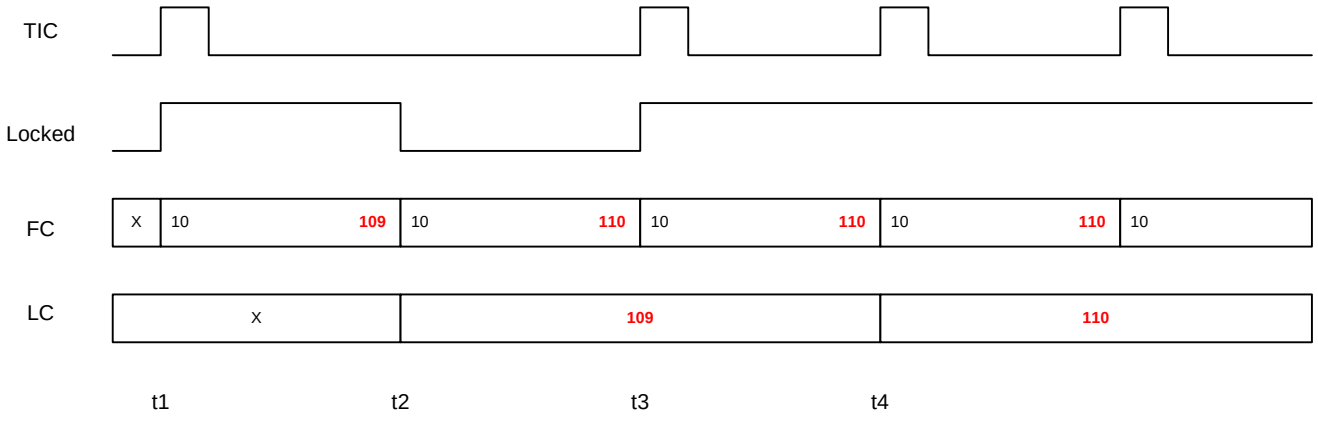
5.2.1 Principle

If the frequency is unchanged at 100Hz, uP_PPS will read LC = 109.
If the frequency changes to 101 Hz unknown to FC, FC still counts from 10 to 109 to generate pulse at 100/101 = 0.99 second. By that time, uP_PPS will write 9 into RC (RC = 9), so FC counts from 9 to 109 to have 101/101 = 1 sec as required.

5.2.2 Implementation

- Whenever TIC available when SV locked, the content of free-running counter FC is latched into counter LC, say Cnt_Value_1. A processor, say uP_PPS (gps_ARM in NTS-200, Net ARM in Xli), reads this Cnt_Value once per second and keeps track on it.
- One second later, uP_PPS reads this counter again, say Cnt_Value_2.
    - If (Cnt_Value_2 = Cnt_Value_1), nothing needs be done
    - If (Cnt_Value_2 = CntValue_1 + 1), uP_PPS writes 9 into Reset Counter RC, so RC changes from 10 to 9

5.2.3 Hardware Implementation



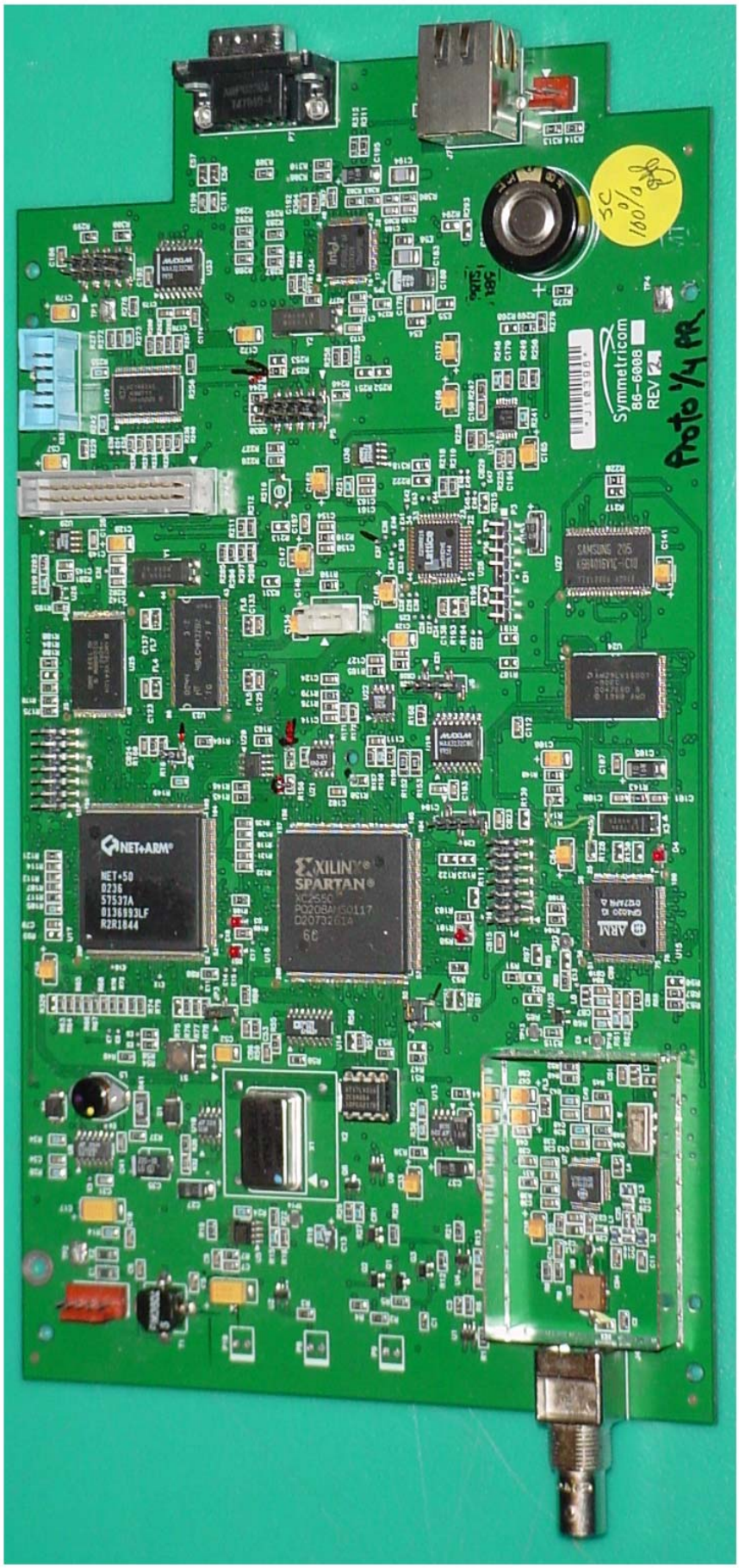
First, FC runs freely for PPS, but not sync'ed to TIC and Locked = 0.

At t1, FC resets to 10, but LC is not latched as Locked(-) = 0, then Locked(+) = 1 dueto TIC

At t2, FC = 109, Locked = 1, so LC latches 109 from FC. Also at t2, there's no TIC, so Locked(+) = 0

At t3, frequency changes and FC = 110 at TIC, but LC is not updated as Locked(-) = 0, then Locked(+) = 1 dueto TIC

At t4, at TIC, FC = 110, LC updates to 110 as Locked(-) = 1.


Symmetricom
86-6008
REV 2A
Proto 1/4 PR
NET+ARM
NET+50
0236
57537A
0136993LF
R2R1844
XILINX
SPARTAN
SAMSUNG 205
ARM