2020-7-23

速度误差计算

 ——

L：探头到挡板距离

T：超声飞行时间

Δt：飞行时间误差

因此由公式，当Δt=1us时，算得ΔV=3.66m/s

\

2020-7-2

1. 修改node\_cfg.h中定义CAN通讯波特率为250K。

2020-6-30

1. 调试单通道measure功能

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Start or stop 上升沿间隔us | 读数 | 计算值‬us | 误差us | 48Mclk |
| 76 | 3567 | 74.3125 | 1.6875 | 46.934 |
| 123 | 5741 | 119.6041‬ | 3.3959 | 46,674 |
|  |  |  |  |  |

2020-6-16

1. 触发AD三个通道同时采样

AD转换时间：

ADC TOTAL CONVERSION TIME = Sample Phase Time (set by SMPLTS + 1) + Hold Phase (1 ADC Cycle) + Compare Phase Time (8-bit Mode = 20 ADC Cycles, 10-bit Mode = 24 ADC Cycles, 12-bit Mode = 28 ADC Cycles) + Single or First continuous time adder (5 ADC cycles + 5 bus clock cycles)

ADC clock=8M ,clockDivide=4,ADC Cycle=2MHz

BUS CLock=4M

T= (12+1+28+5)\*ADC Cycles+5\*BUS Cycles

=46\*1/2000000+5\*1/4000000

=24.25us

2020-6-5

1. 验证AD采样
2. 解决CAN通讯不正常问题。由于单片机晶振由16MHz更换成8MHz，因此CAN通讯模块需要重新配置CLOCK。

修改node\_cfg.h，

**#define** \_CAN\_ClockSpeed (8) //由16改为8M

**#define** \_CAN\_ClockSource (0) //Outside

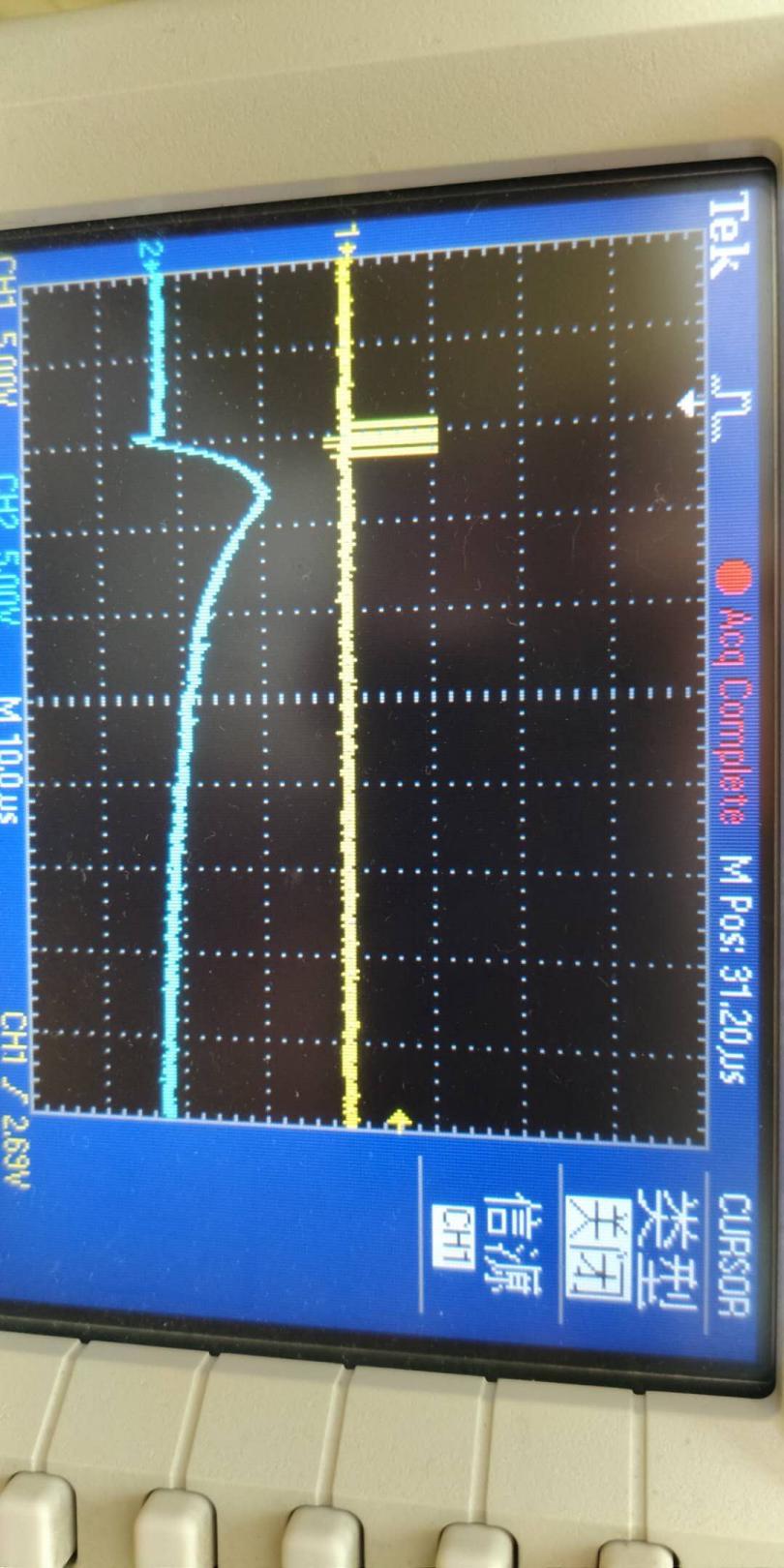
//#define \_CAN\_ClockSource (1) //SYS\_CLK

**#define** \_CAN\_SamplePoint (8125) //81.25%

2020-6-4

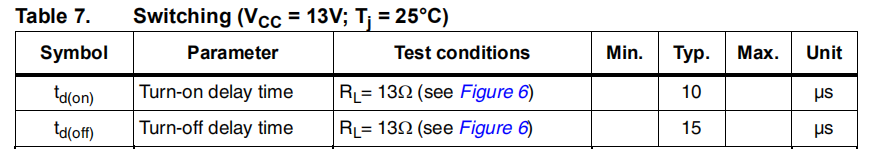
1. 解决软件采样START、STOP之间上升沿间隔小1.1us问题
2. 高压通道测试

问题：VN5160的switch时间不满足，导致HV\_TX上的波形不是1MHz方波。



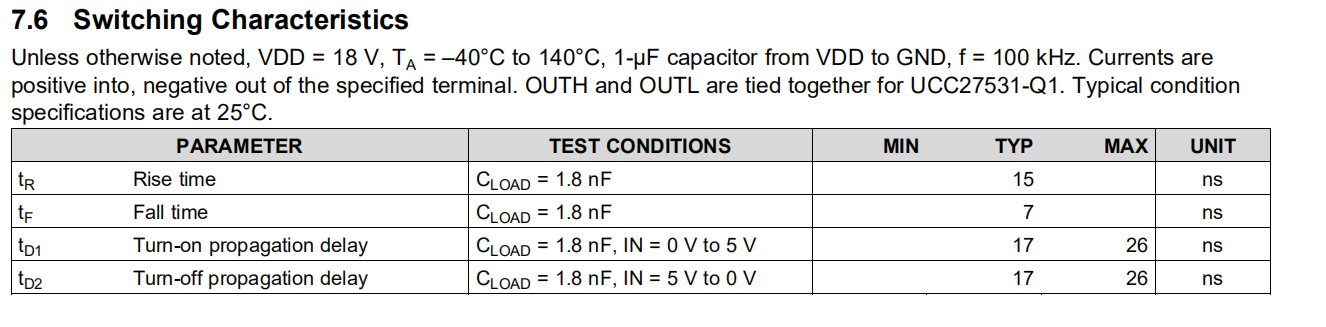
HV\_TX

TDC\_TX1



**VN5160S 开关时间**

需要换成UCC27531-Q1：



**UCC27531-Q1 开关时间**

1. 增加AD采样

VDDA：4V

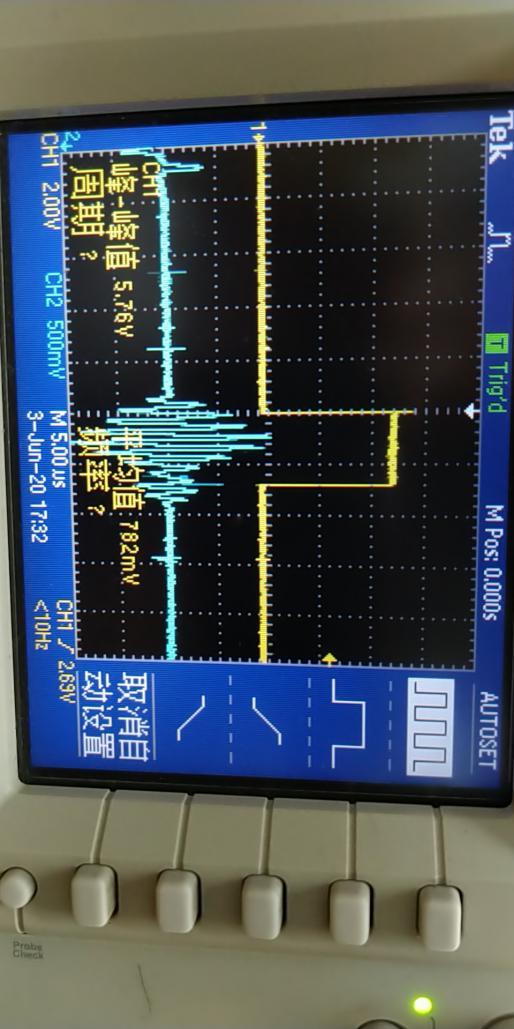
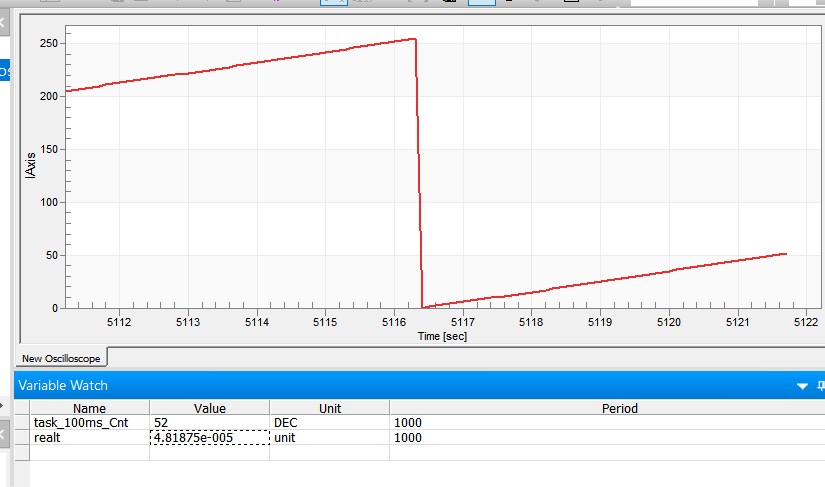
2020-6-3

1. 测试低压通道start、stop捕获

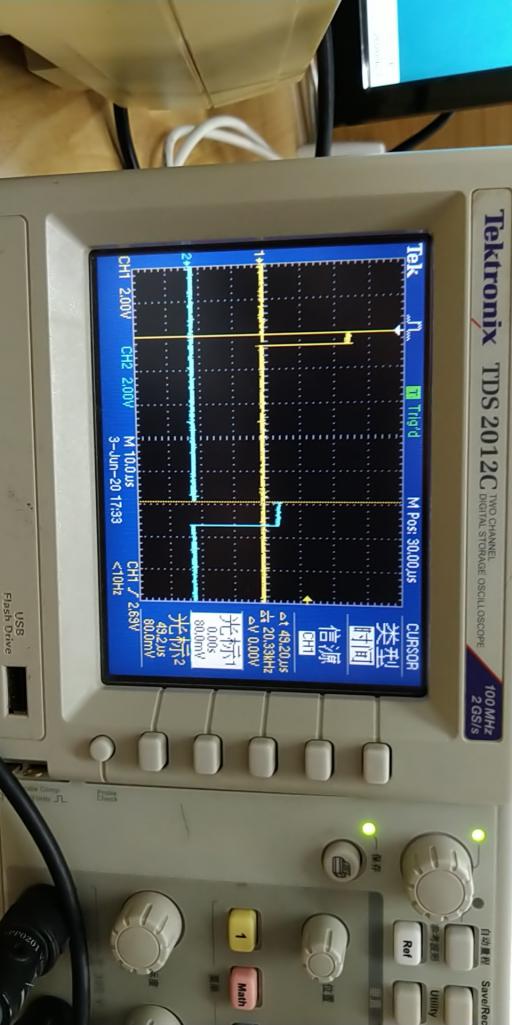
发现问题：a. TDC1000的clk信号用晶振产生没有效果，TDC1000未能工作，用S32K116产生的CLKOUT时钟接入。

b.尝试了两个探头，

当TOF为49.2us, 软件采样START、STOP之间上升沿间隔比示波器测量间隔计算得48.1左右，都小1.1us。



用V3带变压器探头测试，当TOF为163us，软件采样计算得到159us(TOF\_avg=7635,1/48000000\*7635=159.06us)，差4us左右，可见，TOF误差随着读数增大而增大。



2020-6-2

1. 更换CPU为S32K116\_48
2. 调试QLS111 2020.5.25版本

问题：a.电源上电12V拉低到1.1V，电流1mA，单极性TVS 焊反

1. SPI 总线 SDI、SDO 反

2020-6-1

1. 加入TDC1000\_EN、 TDC1000\_RST、 TDC1000\_TRIG初始化
2. 加入触发任务，100ms周期

2020-5-29

1. Change SDK to KSDK 1.1

SDK Path:${ProcessorExpert\_loc}/../../S32DS/S32SDK\_S32K116\_EAR\_1.8.7

1. 加入clockMan1、pin\_mux初始化

Clock 频率设定

FIRC\_Init\_48MHz();

SIRC\_Init\_8MHz();

SOSC\_init\_8MHz();

SYS\_CLK 48MHz——与capture最高速度有关

SCS\_CLK 48MHz

BUS\_CLK 24MHz

SLOW\_CLK 24MHz

Put clock\_Man report here:

1. 加入pin\_mux初始化
2. 修改can初始化

**#define** \_CAN\_ClockSpeed (16) //M

**#define** \_CAN\_ClockSource (0) //Outside

**#define** \_CAN\_SamplePoint (8125) //81.25%

PORTC->PCR[3] |= PORT\_PCR\_MUX(3); /\* Port C3: MUX = ALT3, CAN0\_TX \*/

PORTC->PCR[2] |= PORT\_PCR\_MUX(3); /\* Port C2: MUX = ALT3, CAN0\_RX \*/

uint8\_t **CAN\_Init**(**void**)

1. PIT初始化修改

/\* Clock Src = 3 (FIRCDIV1\_CLK) \*/

/\* TIE0=1: Timer Interrupt Enabled fot Chan 0 \*/

/\* Timeout period: 1k clocks \*/

/\* CHAIN=0: channel chaining is disabled \*/

/\* MODE=0: 32 periodic counter mode \*/

/\* TSOT=0: Timer decrements immediately based on restart \*/

/\* TSOI=0: Timer does not stop after timeout \*/

/\* TROT=0 Timer will not reload on trigger \*/

/\* TRG\_SRC=0: External trigger source \*/

/\* TRG\_SEL=0: Timer chan 0 trigger source is selected \*/

1. SPI初始化

TI设置：

// Reset on, rising edge, 16-bit char bits

// Enable master mode, high (not normal) phase,

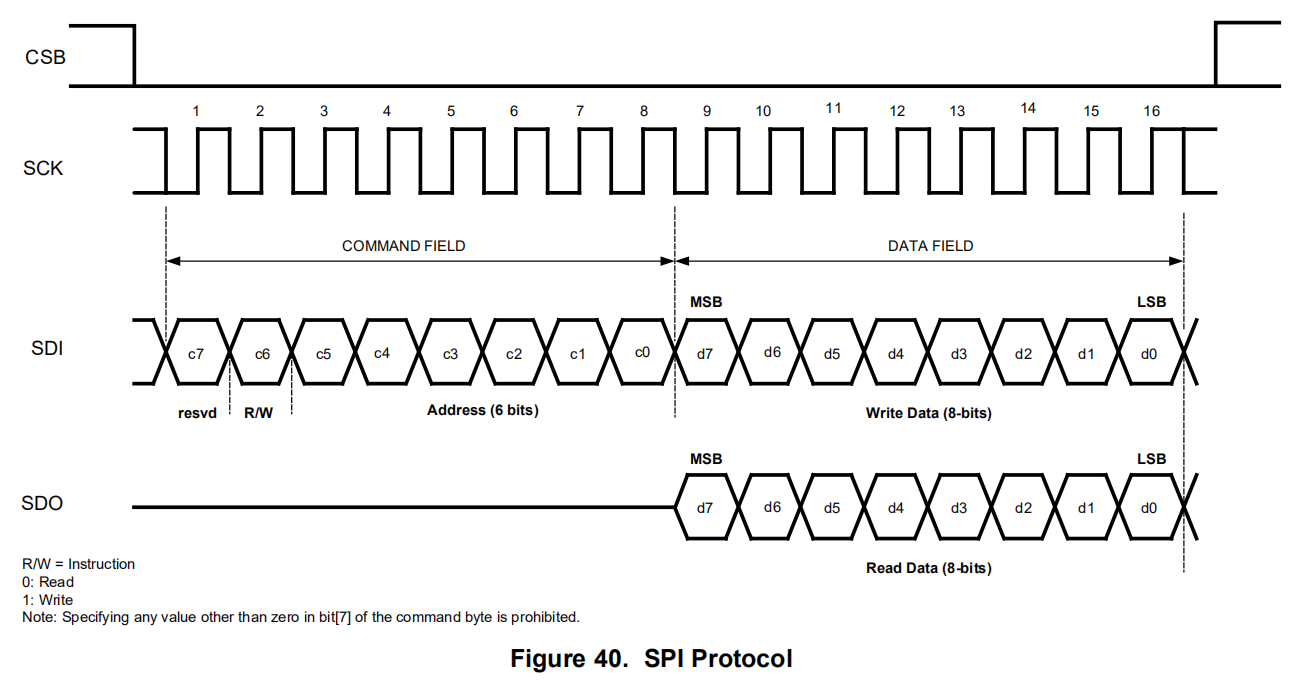
// enable talk, and SPI int disabled.

// vishy: 6MHz spi assuming 60MHz system clock

// vishy: Relinquish SPI from Reset, no loopback, 16-bits

// Set so breakpoints don't disturb xmission

16位，上升沿，时钟高电平，6M，MSB



// 2MHz test cell (yellow)

**#if** 1

**TI\_LMP91400\_SPIWriteReg**(TI\_LMP91400\_CONFIG0\_REG, 0x24); // 4pulses

**TI\_LMP91400\_SPIWriteReg**(TI\_LMP91400\_CONFIG1\_REG, 0x40); // -> 44 to 40 (1stop)

**TI\_LMP91400\_SPIWriteReg**(TI\_LMP91400\_CONFIG2\_REG, 0x04); // tx2 is 04, tx1 is 0

**TI\_LMP91400\_SPIWriteReg**(TI\_LMP91400\_CONFIG3\_REG, 0x0B); // enable blanking, 320mv threshold

**TI\_LMP91400\_SPIWriteReg**(TI\_LMP91400\_CONFIG4\_REG, 0x5F); // 5e (group) -> 1e (edge mode)

**TI\_LMP91400\_SPIWriteReg**(TI\_LMP91400\_TOF1\_REG, 0x80);

**TI\_LMP91400\_SPIWriteReg**(TI\_LMP91400\_TOF0\_REG, 0x1E);

**TI\_LMP91400\_SPIWriteReg**(TI\_LMP91400\_ERROR\_FLAGS\_REG, 0x01);

**TI\_LMP91400\_SPIWriteReg**(TI\_LMP91400\_TIMEOUT\_REG, 0x3B);

**TI\_LMP91400\_SPIWriteReg**(TI\_LMP91400\_CLOCK\_RATE\_REG, 0x01);

**#endif**

// 1MHz test cell (green)

**#if** 0

TI\_LMP91400\_SPIWriteReg(TI\_LMP91400\_CONFIG0\_REG, 0x44); // 4pulses

TI\_LMP91400\_SPIWriteReg(TI\_LMP91400\_CONFIG1\_REG, 0x41); // -> 44 to 40 (1stop)

TI\_LMP91400\_SPIWriteReg(TI\_LMP91400\_CONFIG2\_REG, 0x04); // tx2 is 04, tx1 is 0

TI\_LMP91400\_SPIWriteReg(TI\_LMP91400\_CONFIG3\_REG, 0x0C); // enable blanking, 320mv threshold

TI\_LMP91400\_SPIWriteReg(TI\_LMP91400\_CONFIG4\_REG, 0x5F); // 5e (group) -> 1e (edge mode)

TI\_LMP91400\_SPIWriteReg(TI\_LMP91400\_TOF1\_REG, 0x40);

TI\_LMP91400\_SPIWriteReg(TI\_LMP91400\_TOF0\_REG, 0x1E);

TI\_LMP91400\_SPIWriteReg(TI\_LMP91400\_ERROR\_FLAGS\_REG, 0x01);

TI\_LMP91400\_SPIWriteReg(TI\_LMP91400\_TIMEOUT\_REG, 0x23);

TI\_LMP91400\_SPIWriteReg(TI\_LMP91400\_CLOCK\_RATE\_REG, 0x01);

**#endif**

2020-5-28

SDK 1.87加入S32K116\_App\_