

**Delphi Electronics & Safety**

**Powertrain Gas Product Line**

**Software Manual Test Report**

**For**

**MT22.3 ETC**

**Chery HWIO 4Cyl ECM**

**Date: 10 April 2015**

Table of Contents

[1. Check Complex IO Fuel Logic 1](#_Toc416102281)

[1.1 Fuel Inject Simultaneous 1](#_Toc416102282)

[1.2 Fuel Post Inject 2](#_Toc416102283)

[2. Check Spark Logic 6](#_Toc416102284)

[2.1 The dwell time of follow up main spark 6](#_Toc416102286)

[2.2 The break time of follow up sparks 7](#_Toc416102287)

[2.3 The follow up sparks 11](#_Toc416102288)

[3. Stack depth measurement 15](#_Toc416102289)

[4. Memory resource measurement 16](#_Toc416102290)

[5. Check ECM Re-Programming 17](#_Toc416102291)

Because that the OPAL\_RT could not capture the Inject and spark that with multi-pulse, so the post inject and the multiple spark could only be manual measured by oscilloscope. And the LLD software version is chery\_mt22.3\_fix@115\_fix\_security\_access\_issue\_20150402.

# Check Complex IO Fuel Logic

## 1.1 Fuel Inject Simultaneous

This test verifies fuel logic works for simultaneous fuel delivery at start up as per cylinder number.

**Test procedure:**

1. Keep the RPM = 800r/m ( crank\_sig.engine\_rpm=3200 )
2. Key off/Key on, verify if the startup simultaneous fuel are delivered immediately at the falling edge of 3rd tooth. The waveform sequence is the same as follows.

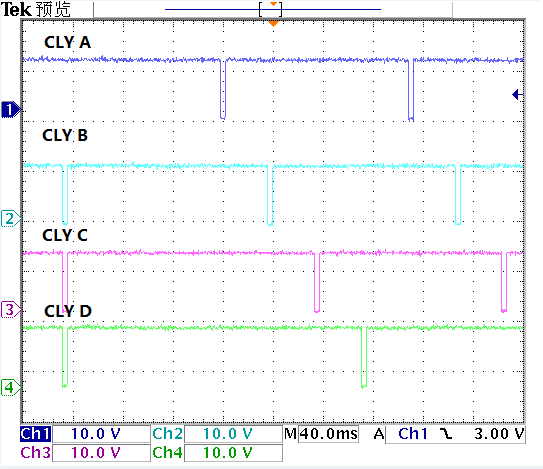


Fig1.1.1 The startup injection sequence when cam low

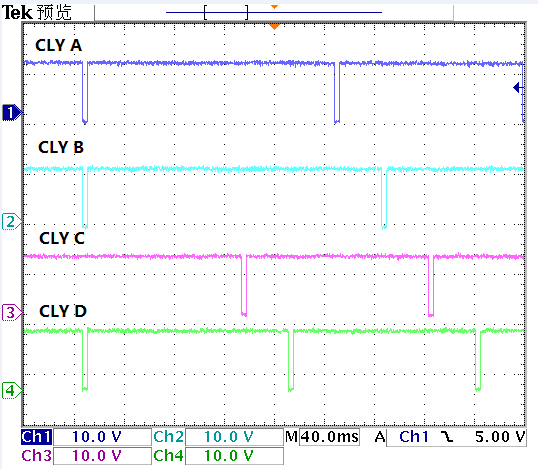


Fig 1.1.2 The startup injection sequence when cam high

**Test Result:** **PASS**

## 1.2 Fuel Post Inject

This test verifies fuel logic produces post injection fuel pulse.

Test code:

void HLS\_syn(void)

{

inj\_sig[INJ\_CHANNEL\_A].inj\_time = 5000;

inj\_sig[INJ\_CHANNEL\_B].inj\_time = 5000;

inj\_sig[INJ\_CHANNEL\_C].inj\_time = 5000;

inj\_sig[INJ\_CHANNEL\_D].inj\_time = 5000;

inj\_sig[INJ\_CHANNEL\_A].B\_post\_inj = 1;

inj\_sig[INJ\_CHANNEL\_B].B\_post\_inj = 1;

inj\_sig[INJ\_CHANNEL\_C].B\_post\_inj = 1;

inj\_sig[INJ\_CHANNEL\_D].B\_post\_inj = 1;

inj\_sig[INJ\_CHANNEL\_A].post\_inj\_time = 2000;

inj\_sig[INJ\_CHANNEL\_B].post\_inj\_time = 2000;

inj\_sig[INJ\_CHANNEL\_C].post\_inj\_time = 2000;

inj\_sig[INJ\_CHANNEL\_D].post\_inj\_time = 2000;

inj\_sig[INJ\_CHANNEL\_A].inj\_end\_angle = 30;

inj\_sig[INJ\_CHANNEL\_B].inj\_end\_angle = 30;

inj\_sig[INJ\_CHANNEL\_C].inj\_end\_angle = 30;

inj\_sig[INJ\_CHANNEL\_D].inj\_end\_angle = 30;

}

**Test procedure**:

1. Keep rpm= 1000r/m, ( crank\_sig.engine\_rpm=4000 )
2. Keep fuel pulse width 5 msec, inj\_sig[X].inj\_time = 5000, X=A,B,C,D
3. Set the post injection time 2ms, (inj\_sig[X].post\_inj\_time=2000, X=A,B,C,D )
4. Enable the post injection, (inj\_sig[X].B\_post\_inj = true, X=A,B,C,D )
5. Change the Fuel inject pulse end angle( inj\_sig[INJ\_CHANNEL\_A].inj\_end\_angle )

**Test Result:**

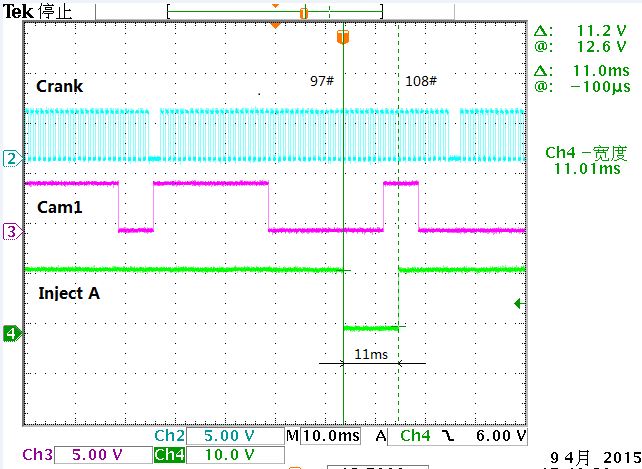


Fig 1.2.1 inj\_sig[INJ\_CHANNEL\_A].inj\_end\_angle =30

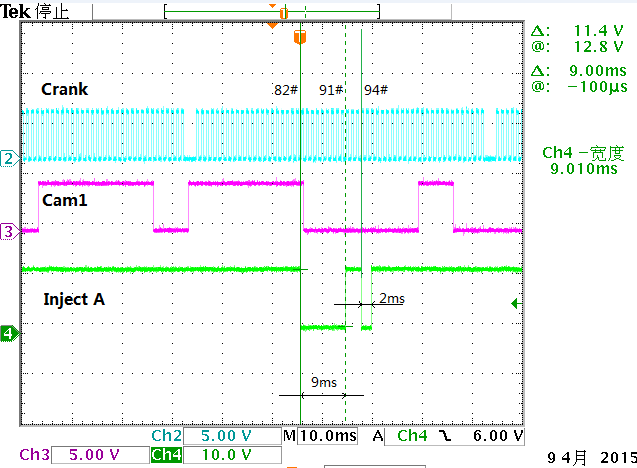


Fig 1.2.2 inj\_sig[INJ\_CHANNEL\_A].inj\_end\_angle =70

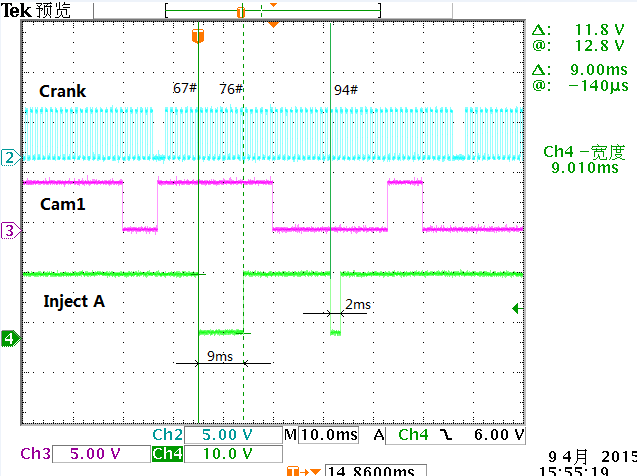


Fig 1.2.3 inj\_sig[INJ\_CHANNEL\_A].inj\_end\_angle =100

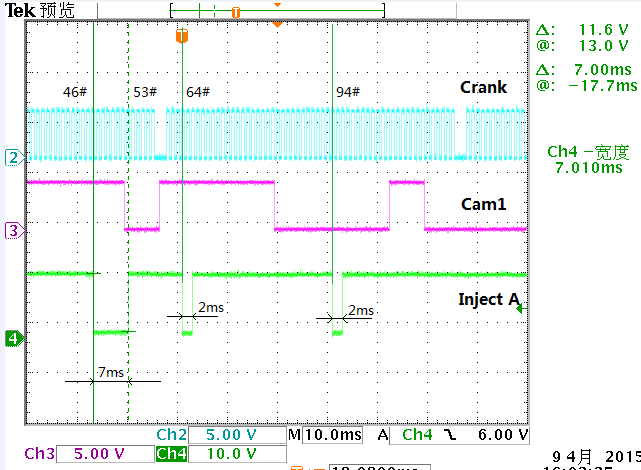


Fig 1.2.4 inj\_sig[INJ\_CHANNEL\_A].inj\_end\_angle =150

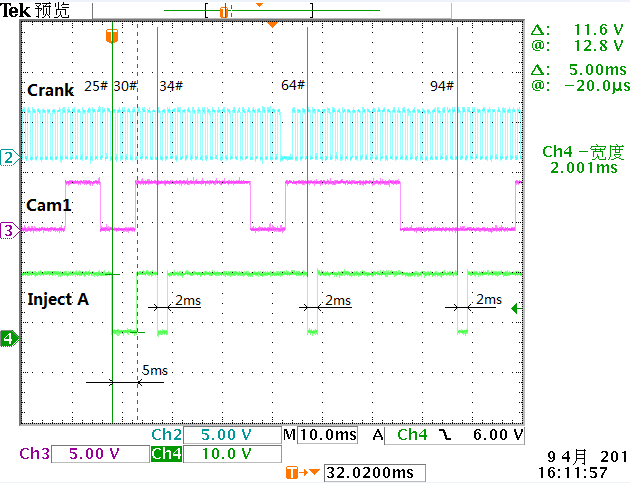


Fig 1.2.5 inj\_sig[INJ\_CHANNEL\_A].inj\_end\_angle =200

**Test Result:** **PASS**

# Check Spark Logic

### 

## 2.1 The dwell time of follow up main spark

**Test procedure:**

1. Keep rpm= 1000r/m, ( crank\_sig.engine\_rpm=4000 )
2. Set ign\_sig[A].enable =1, Set ign\_sig[C].enable =0
3. Set ign\_sig[A]. ign\_angle = 0
4. Set ign\_sig[A].follow\_up\_sparks =1
5. Set ign\_sig[A].dwell\_time =4000
6. Set ign\_sig[A].break\_time\_of\_follow\_up\_sparks =500
7. Change the follow up spark dwell time by ign\_sig[A].dwell\_time\_of\_follow\_up\_sparks, as follow.

**Test Result:**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Expected follow up Spark dwell time(us)** | **Actual follow up Spark dwell time(us)** | **Result** |
| 1 | 100 | 104 | Pass |
| 2 | 200 | 198 | Pass |
| 3 | 500 | 511 | Pass |
| 4 | 1000 | 1011 | Pass |
| 5 | 2000 | 2012 | Pass |
| 6 | 5000 | 5014 | Pass |
| 7 | 10000 | 10020 | Pass |

## 

Fig2.1.1 ign\_sig[A].dwell\_time\_of\_follow\_up\_sparks =100

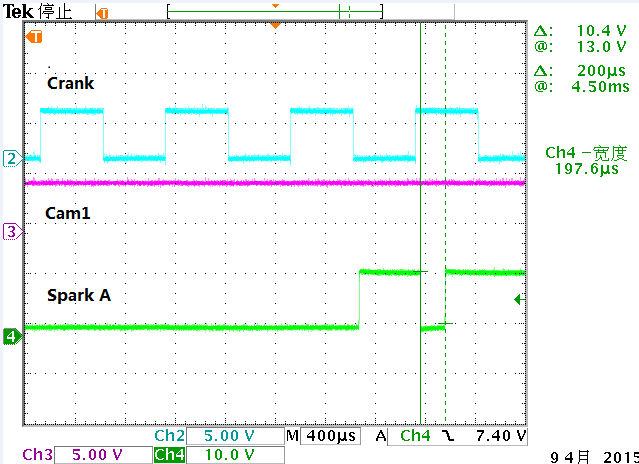


Fig 2.1.2 ign\_sig[A].dwell\_time\_of\_follow\_up\_sparks =200

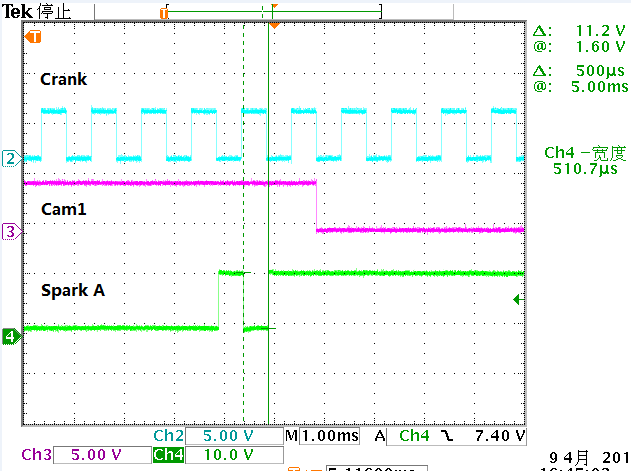


Fig 2.1.3 ign\_sig[A].dwell\_time\_of\_follow\_up\_sparks =500

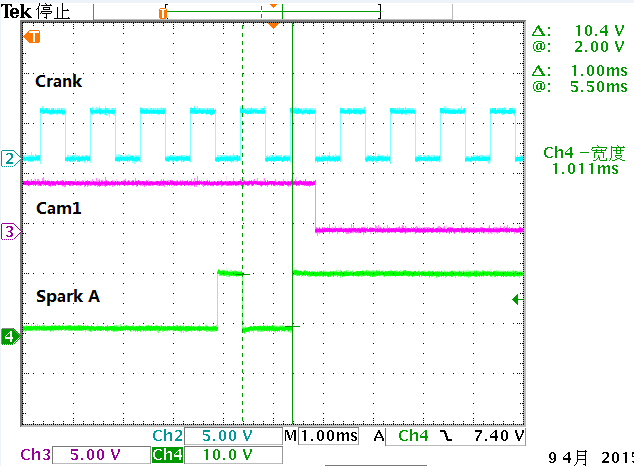


Fig 2.1.4 ign\_sig[A].dwell\_time\_of\_follow\_up\_sparks =1000

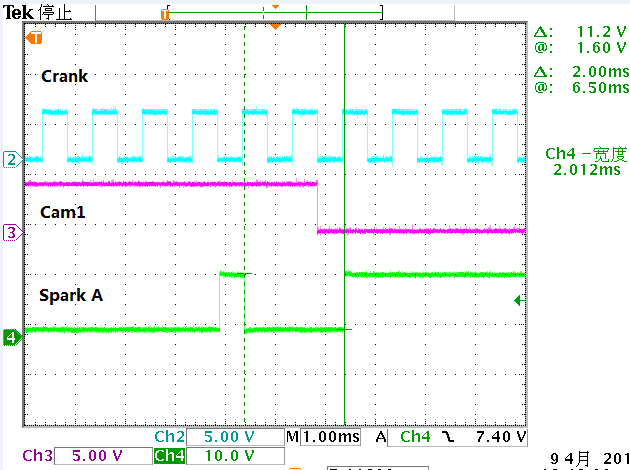


Fig 2.1.5 ign\_sig[A].dwell\_time\_of\_follow\_up\_sparks =2000

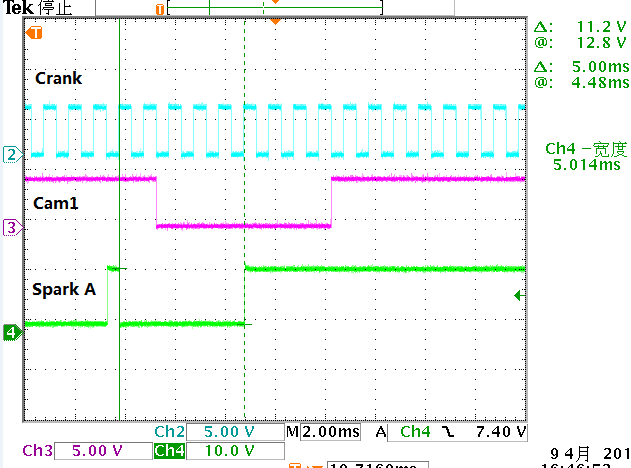


Fig 2.1.6 ign\_sig[A].dwell\_time\_of\_follow\_up\_sparks =5000

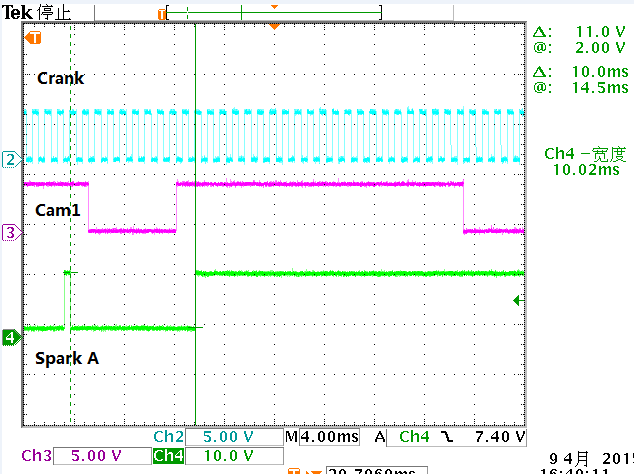


Fig 2.1.7 ign\_sig[A].dwell\_time\_of\_follow\_up\_sparks =10000

## 2.2 The break time of follow up sparks

**Test procedure:**

1. Keep rpm= 1000r/m, ( crank\_sig.engine\_rpm=4000 )
2. Set ign\_sig[A].enable =1, Set ign\_sig[C].enable =0
3. Set ign\_sig[A]. ign\_angle = 0
4. Set ign\_sig[A].follow\_up\_sparks =1
5. Set ign\_sig[A].dwell\_time =4000
6. Set ign\_sig[A].dwell\_time\_of\_follow\_up\_sparks =1000
7. Change ign\_sig[A].break\_time\_of\_follow\_up\_sparks

**Test Result:**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Expected break time of follow up sparks(us)** | **Actual break time of follow up sparks (us)** | **Result** |
| 1 | 100 | 84 | Pass |
| 2 | 200 | 178 | Pass |
| 3 | 500 | 491 | Pass |
| 4 | 1000 | 990 | Pass |
| 5 | 2000 | 1992 | Pass |
| 6 | 5000 | 4992 | Pass |

## Notes: The no.1 and no.2 is small break time test, and the delta break time is stable.

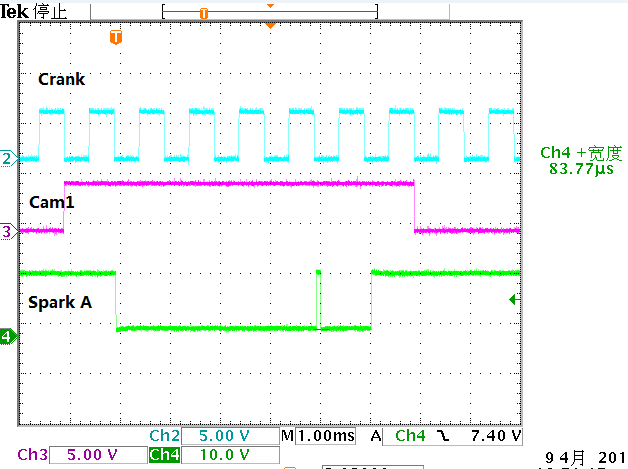


Fig 2.2.1 ign\_sig[A].break\_time\_of\_follow\_up\_sparks =100

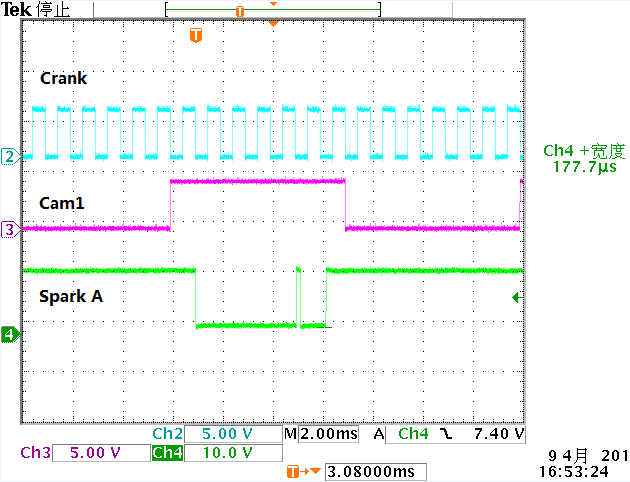


Fig 2.2.2 ign\_sig[A].break\_time\_of\_follow\_up\_sparks =200

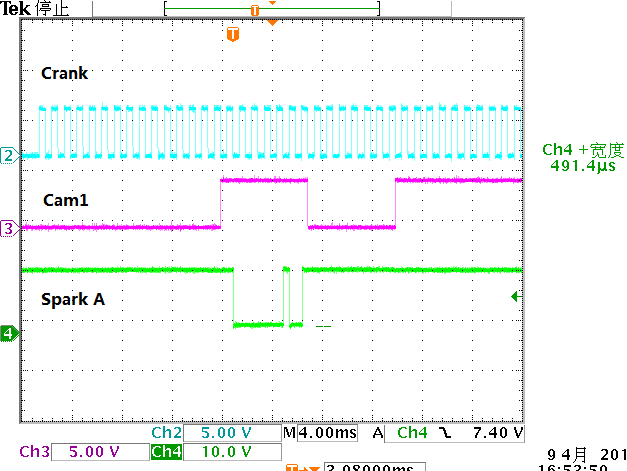


Fig 2.2.3 ign\_sig[A].break\_time\_of\_follow\_up\_sparks =500

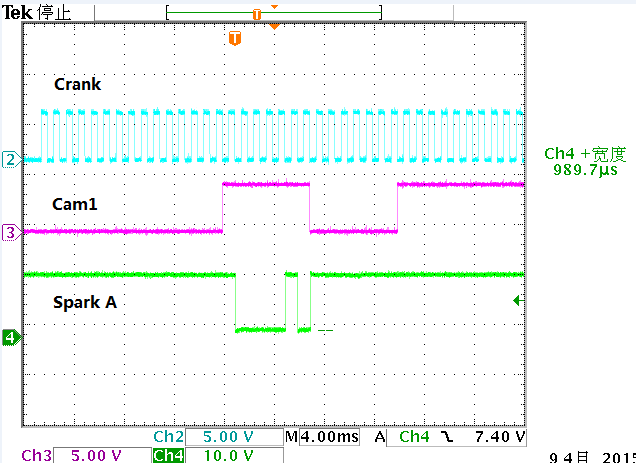


Fig 2.2.4 ign\_sig[A].break\_time\_of\_follow\_up\_sparks =1000

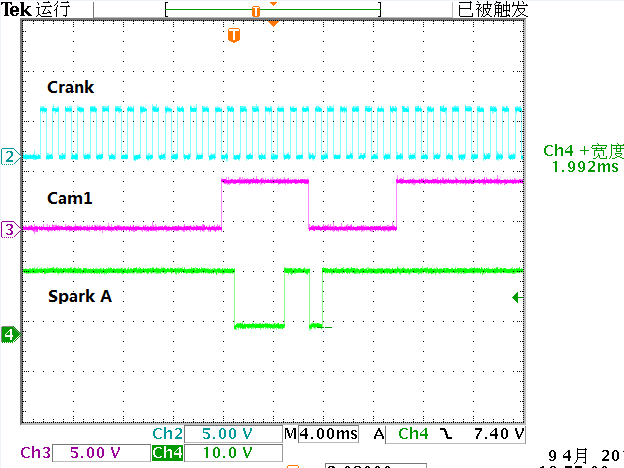


Fig 2.2.5 ign\_sig[A].break\_time\_of\_follow\_up\_sparks =2000

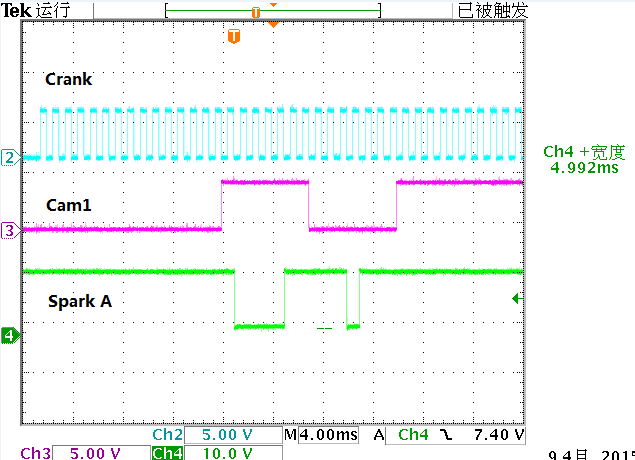


Fig 2.2.6 ign\_sig[A].break\_time\_of\_follow\_up\_sparks =5000

## 2.3 The follow up sparks

**Test procedure:**

1. Keep rpm= 1000r/m, ( crank\_sig.engine\_rpm=4000 )
2. Set ign\_sig[A].enable =1, Set ign\_sig[C].enable =0
3. Set ign\_sig[A]. ign\_angle = 0, x =A, B, C, D
4. Set ign\_sig[A].dwell\_time =4000
5. Set ign\_sig[A].dwell\_time\_of\_follow\_up\_sparks =1000
6. Set ign\_sig[A].break\_time\_of\_follow\_up\_sparks =500
7. Change ign\_sig[A].follow\_up\_sparks

**Test Result:**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Expected follow up sparks**  **(count)** | **Actual follow up Spark**  **(count)** | **Result** |
| 1 | 0 | 0 | Pass |
| 2 | 1 | 1 | Pass |
| 3 | 2 | 2 | Pass |

# 

Fig 2.3.1 ign\_sig[A].follow\_up\_sparks =0

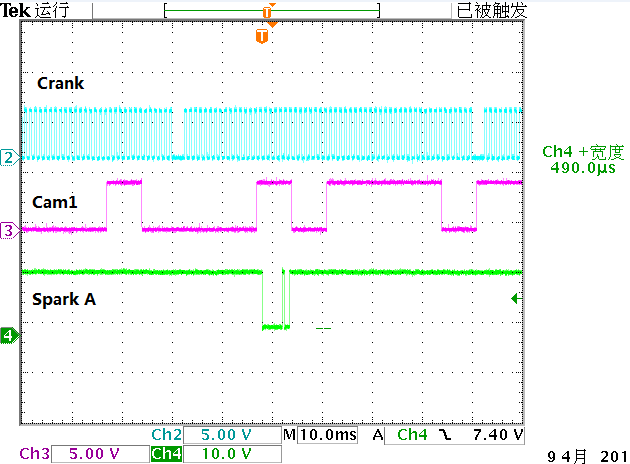


Fig 2.3.2 ign\_sig[A].follow\_up\_sparks =1

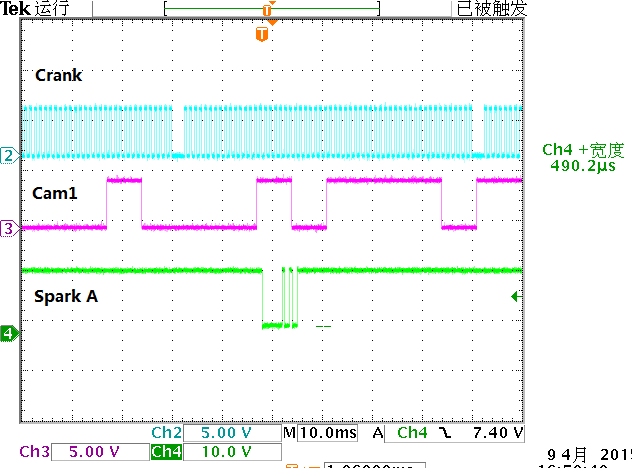


Fig 2.3.3 ign\_sig[A].follow\_up\_sparks =2

# Stack depth measurement

**Test procedure:**

-Integrate the DELPHI LLD software with CHERY HLS.

-Connect the Debugger to the ECM

- Reset the processor, block fill memory address range 0x4000F000 to 0x4000FFFF with value 0x00.

- Vary the engine Speed from 0 to 6000 rpm.

- Record the lowest RAM memory location (range 0x4000F000 to 0x4000FFFF) which content is not 0X00.

(Note: The stack grew from higher to lower address. Top of stack (TOS) is at address 0x4000FFFF).

**Test Result: 1263(**0x4000FB10 to 0x4000FFFF)

# Memory resource measurement

Check all memory ranges and maximum available memory of each range in the CPU memory usage page.

Identify as a risk if any of the banks/ranges exceeded 90% usage.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Memory Type** | | | **Total Size**  **(Bytes)** | **Used Size**  **(Byte)** | **Present**  **(%)** |
| ROM | App\_code | | 1244K | 673.9K(0XA87B0) | 54.2% |
| Cal | | 128K | 83K(0X14C06) | 64.8% |
| RAM | App\_ram | Data | 52K | 2.3K(0x93C) | 53.3% |
| Bss | 14.9K(0X3BCC) |
| Hlsbss | 10.5K(0X2A00) |
| Nc\_nvram | | 6.75K(0X1B00) | 5k(0x1414) | 74.4% |
| Fcm\_ram | | 640(0X280) | 44(0X2C) | 6.9% |

# Check ECM Re-Programming

Verify the (non-instrumented) ECM can be re-programmed using the flashtools and INCA. Do this test twice.

**Test Result: Pass**