Use of PLD’s in Fuze Safety Critical Circuits

Bob Hubal
Fuze Division, FPAT
METC, ARDEC
• This paper will present my views and recent experience working with the FESWG to study the use of Programmable Logic Devices in fuzes.
• Present data gathered by the FESWG and other sources will be presented.
  ❖ Being a fuze designer and also a member of the Army Fuze Safety Board gives me a unique viewpoint on this issue.
• Programmable Logic Devices (PLD) are essential tools to the fuze designer

• PLD’s enable advanced fuzes to perform complex and precisely timed functions, measure environments, communicate with other subsystems, etc

• PLD’s are used in large percentage of electronic fuzes
Issue

• PLD’s that use charge based memory cell technology are rapidly replacing PLD’s that use fixed “fusible type” memory cells
• In particular newer model PLD’s which have improved performance such as reduced power consumption are only available in the charged based memory cell technology
• PLD vendors claim long memory retention time for charged based memory cells (ex: 40 years)
• Basic question for the fuze safety community – Can we rely on these devices in safety critical fuze circuits and if so what are the guidelines for their use?
• To date only fusible type FPGA’s or OTP type microcontrollers have been approved by Fuze Safety Boards.

• The FESWG created a subcommittee in 2009 to study the use of PLD’s in fuze circuits.
  – Main question to be answered:
    “Is there an acceptable way to use PLD’s that use charge based memory technology in safety critical fuze circuits?”
Fuze Life and PLD Memory Loss

Must be safe

Ship, Store and Transport

Mfg

Fuze

( up to 20 Years + extreme environments)

Field Mission

Operation

Memory

Good

(In circuit tests)

Is memory still good???
Any fuze part can fail for a number of reasons and in general fuzes are designed to stay safe if a single point failure occurs.

However if there is an inherent memory loss failure mechanism in a PLD, a significant number of fuzes could experience failures during long term storage.

The software code would fail randomly and the PLD’s performance could become unpredictable.

**Overall system safety would be degraded.**
FESWG Subcommittee Initiatives

• Tasked Sandia NL under the DOD/DOE TCG program to assess the validity of vendors memory retention claims
• Studied potential circuit techniques and guidelines that could mitigate the effects of PLD memory loss on safety
• Initiated an update of the FESWG Logic devices guidelines
• Sample PLD’s were tested by Sandia NL’s in an attempt to verify vendors memory retention claims

• Two flash microcontrollers and an FPGA’s were selected for test

• The following tests were conducted:
  – Unpowered HAST/130C
  – Temperature Cycling@-55 to +150C
  – High Temp Storage @ 150C
Summary of Data Retention Test Results

<table>
<thead>
<tr>
<th></th>
<th>HAST unpowered</th>
<th>Temp Cycling -55/+150C</th>
<th>Data Retention</th>
</tr>
</thead>
<tbody>
<tr>
<td>FPGA</td>
<td>2 wire bond failures in 300-400 hrs</td>
<td>0 Failures 1500/2000 cycles</td>
<td>0 failures 3000 hrs@150C</td>
</tr>
<tr>
<td>Micro 1</td>
<td>0 failures in 350 hrs</td>
<td>0 failures 1000/1500 cycles</td>
<td>0 failures 3000 hrs@150C</td>
</tr>
<tr>
<td>Micro 2</td>
<td>N/A</td>
<td>N/A</td>
<td>0 failures 8000 hrs@150C</td>
</tr>
</tbody>
</table>

No memory cell failures!
Memory Checking

Mfg

Ship, Store and Transport

Field Mission

Power-up  Operation

Fuze

(>Years<)

Memory

Good

(In circuit tests)

Is memory still good???
If we can determine this here we can failsafe the system if the memory is corrupted.
PLD has to derive the right combination based on checking it’s memory.
This value is not resident in memory
The memory check is robust and the value is unlikely to be generated by mistake.
Purpose:
Checks program memory against an external “coded word” (a.k.a. key).
Re-programmability feature shall be defeated robustly (Service-review required).
Hubal Key acts only as a check for the integrity of EPROM/EEPROM/Flash memory.
Hubal Key does not check hardware or processing functions.
Lines between µC and Hubal Key are dedicated and shall not be used for any other purposes, including monitoring.
Memory integrity check shall be run upon the application of power and at the start of all arming processes.
Concept for In-Circuit Check of PLD Memory

- Test code
- Tactical

1. Perform 16 Bit CRC check
2. Add in offset value
3. Send 16 bits to I/O port
4. Shift out 16 bits to Memory Checker Ckt
5. Memory checker Ckt: Compares micro’s 16 bits to stored Hubal “key” code (65k possible combos)

Read program memory
Alternate memory checking scheme — Mirror tactical code and check against itself

Tactical code + Complement of tactical code = 0

Test code

Build 16 bit code word based on results

Advantage: Fast – simple processing

Disadvantage: doubles memory size
Drawbacks and Limitations of Memory checking scheme

- Takes extra electronic parts to implement.
- More applicable to micro’s, fpga’s not as straightforward.
  - Can be implemented in a custom IC or by a small discrete SMD ckt
- It takes time to conduct the memory check - this may cause a problem for some systems that have to arm quickly (ex: APS & high velocity close engagement rounds).
- Memory loss still results in a fuze dud
Example 5. Assembly Language for CRC-16 Using a Lookup Table

crc_lo data 40h ; any direct address is okay
crc_hi data 41h
tmp data 42h

;---------------------------------------------------------------------------
; CRC16 subroutine.
; - accumulator is assumed to have byte to be crc'ed
; - three direct variables are used, tmp, crc_hi and crc_lo
; - crc_hi and crc_lo contain the CRC16 result
; - this CRC16 algorithm uses a table lookup
;---------------------------------------------------------------------------
crc16:
  xrl a, crc_lo ; create index into tables
  mov tmp, a ; save index
  push dph ; save dptr
  push dpl ;
  mov dptr, #crc16_tablo ; low part of table address
  movc a, @a+dptr ; get low byte
  xrl a, crc_hi ;
  mov crc_lo, a ; save of low result
  mov dptrl, #crc16_tablo ; high part of table address
  mov a, tmp ; index
  movc a, @a+dptr ;
  mov crc_hi, a ; save high result
  pop dpl ; restore pointer
  pop dph ;
  ret ; all done with calculation

Sample Time to Conduct a CRC check

Sample computation time calc. for 16 bit crc check:

Assumptions

4 Mhz clock (typ internal clock freq.)

4 clock cycles /instruction – typ for Microchip parts

4k of tactical code

15 instructions for 2 bytes of memory check

1 usec per intruction x 15 instructions x 4k/2 =

30 msec +

Note: A lookup table with crc values must be stored in memory for this program
• SOPM ESAD Safety Architectures makes use of a charged based device due to size constraints.

• Implementing CRC 16 verification of stored memory
  – Charged based device interfaces with dedicated discrete circuitry via two pins.
  – Compares expected hardwired known CRC bit-by-bit against device’s memory.
  – Check occurs first, at power up.
  – Power shut down of the device occurs if:
    • Mis-match occurs
    • More than 16 bits are clocked out
    • Less than 16 bits are clocked in fixed time frame

• Active components includes a combination of shift registers, counters, and timers.

• Minimal board real estate impact.
  – Projected to be not more than 260 mm² or 0.4 in² of layout space.
Summary

• The FESWG Logic Devices tech manual has been updated to address use of charged based PLD’s and include memory checking
  • Looking for feedback from the fuze industry
• More study of PLD’s needed
  • Additional memory retention tests to increase confidence level
  • Develop an understanding of failure mechanisms and possible screening techniques
• Almost certainly new technologies will emerge that the fuze safety community will have to deal with

Today

**PLD’s**

Tomorrow

???