GENERAL DYNAMICS
Canada

BSDCAN 2007

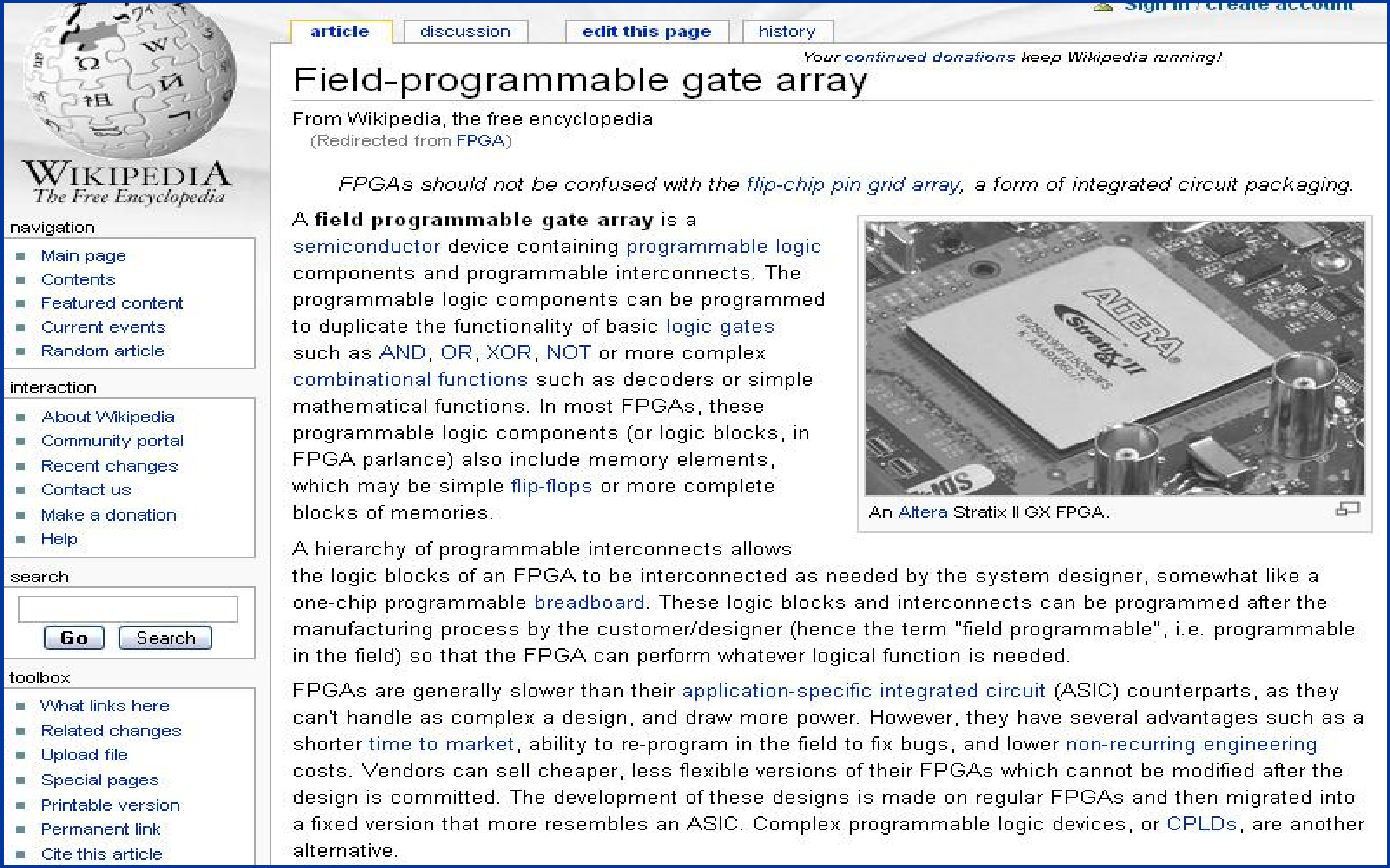
Howard Harvey
Howard.Harvey@gdcanada.com

May 2007

Introduction

- General Dynamics is developing a series of ruggedized military communications products based on NetBSD and Xilinx FPGA's

FPGA Definition



Your continued donations keep Wikipedia running!

Field-programmable gate array

From Wikipedia, the free encyclopedia

(Redirected from FPGA)

FPGAs should not be confused with the flip-chip pin grid array, a form of integrated circuit packaging.

A **field programmable gate array** is a semiconductor device containing programmable logic components and programmable interconnects. The programmable logic components can be programmed to duplicate the functionality of basic logic gates such as AND, OR, XOR, NOT or more complex combinational functions such as decoders or simple mathematical functions. In most FPGAs, these programmable logic components (or logic blocks, in FPGA parlance) also include memory elements, which may be simple flip-flops or more complete blocks of memories.

An Altera Stratix II GX FPGA.

A hierarchy of programmable interconnects allows the logic blocks of an FPGA to be interconnected as needed by the system designer, somewhat like a one-chip programmable breadboard. These logic blocks and interconnects can be programmed after the manufacturing process by the customer/designer (hence the term "field programmable", i.e. programmable in the field) so that the FPGA can perform whatever logical function is needed.

FPGAs are generally slower than their application-specific integrated circuit (ASIC) counterparts, as they can't handle as complex a design, and draw more power. However, they have several advantages such as a shorter time to market, ability to re-program in the field to fix bugs, and lower non-recurring engineering costs. Vendors can sell cheaper, less flexible versions of their FPGAs which cannot be modified after the design is committed. The development of these designs is made on regular FPGAs and then migrated into a fixed version that more resembles an ASIC. Complex programmable logic devices, or CPLDs, are another alternative.

FPGA Development

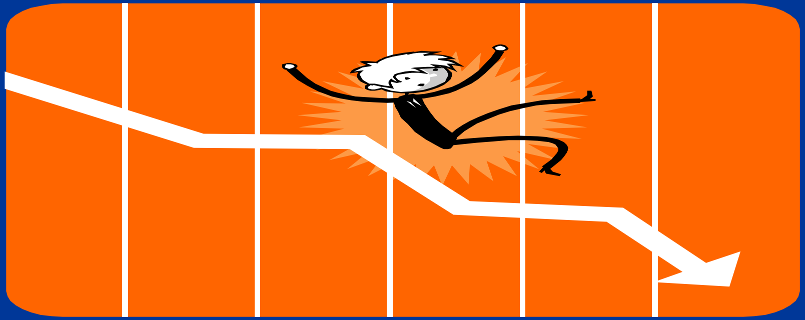
- Why would one ever want to develop with an FPGA device?
  - FPGA's are the current "Kewl" technology?
  - Not really - anyone remember "ASIC"s ?

FPGA Development – Good things

- Hardware fixes no longer require a soldering iron, a magnifying glass and a steady hand
- Extensive simulation prior to deployment
  - Good chance it will “just work” ™ ®
- Chipscope – DTrace for hardware
- Embeddable DSP blocks for FPGA available

The “Bad” OLD Days

- Previously, General Dynamics developed hardware, tossed it over the wall to the firmware team for deployment who tossed it over the wall for the final applications development, classic "waterfall" development model.



The Bad Old Days (continued…)

- Ada runtimes on proprietary hardware
- Running flat mode – no MMU turned on
    - Count on language features to find bugs, and not hardware memory protection
- Hardware problems discovered only months after the hardware was built, expensive re-spins, nasty relations between hardware and software camps.

..still The "Bad" OLD Days

- Military development world is not immune to the pressures of the commercial world
- They want it
    - Smaller
    - Faster
    - Cheaper

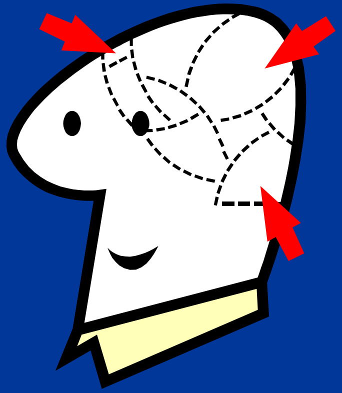
Did I mention "Cheaper and Faster"

Why use an FPGA?

- FPGA's are the re-instatement of the "Natural Order"?
- Hardware that takes it's marching orders from software, what could be more natural than that?

Developer Mindset Changes

- It means your software developers become more “hardware aware”
- Your hardware developers become more "software aware”
- FPGA's are software - there is no real hardware anymore.....
- Hardware-Software==Firmware?

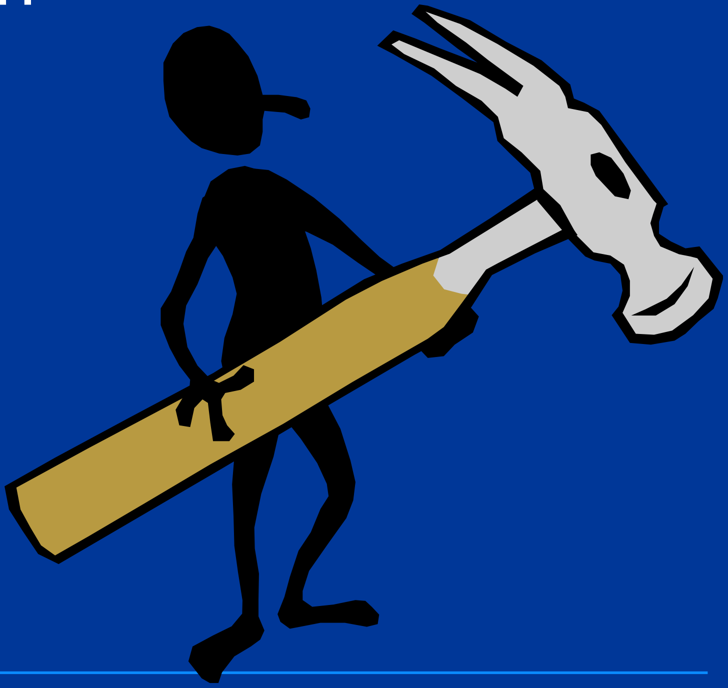


Geletta's Law

- Geletta's Law of Product Development Velocity
  - It's not how fast you can build it,
  - It's how fast you can -*FIX*- it

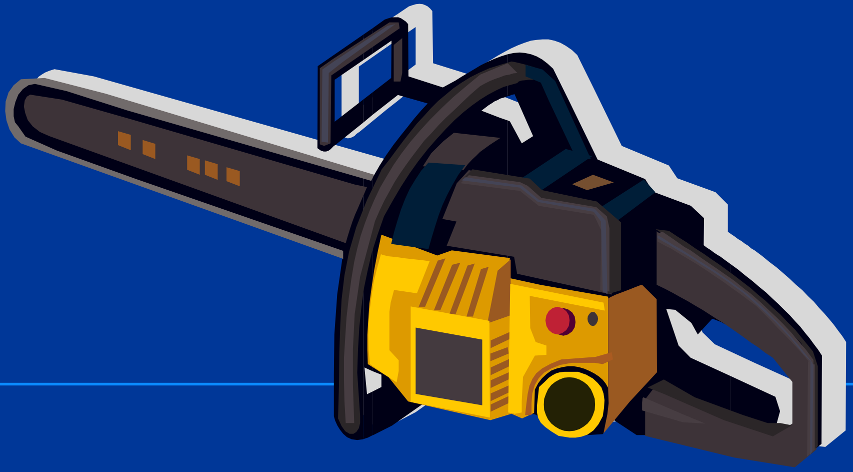
If you have to fix it ….

- Make sure you have the right tools,
  - they had best be "sharp tools"
- GDB -> DDD via an abatron



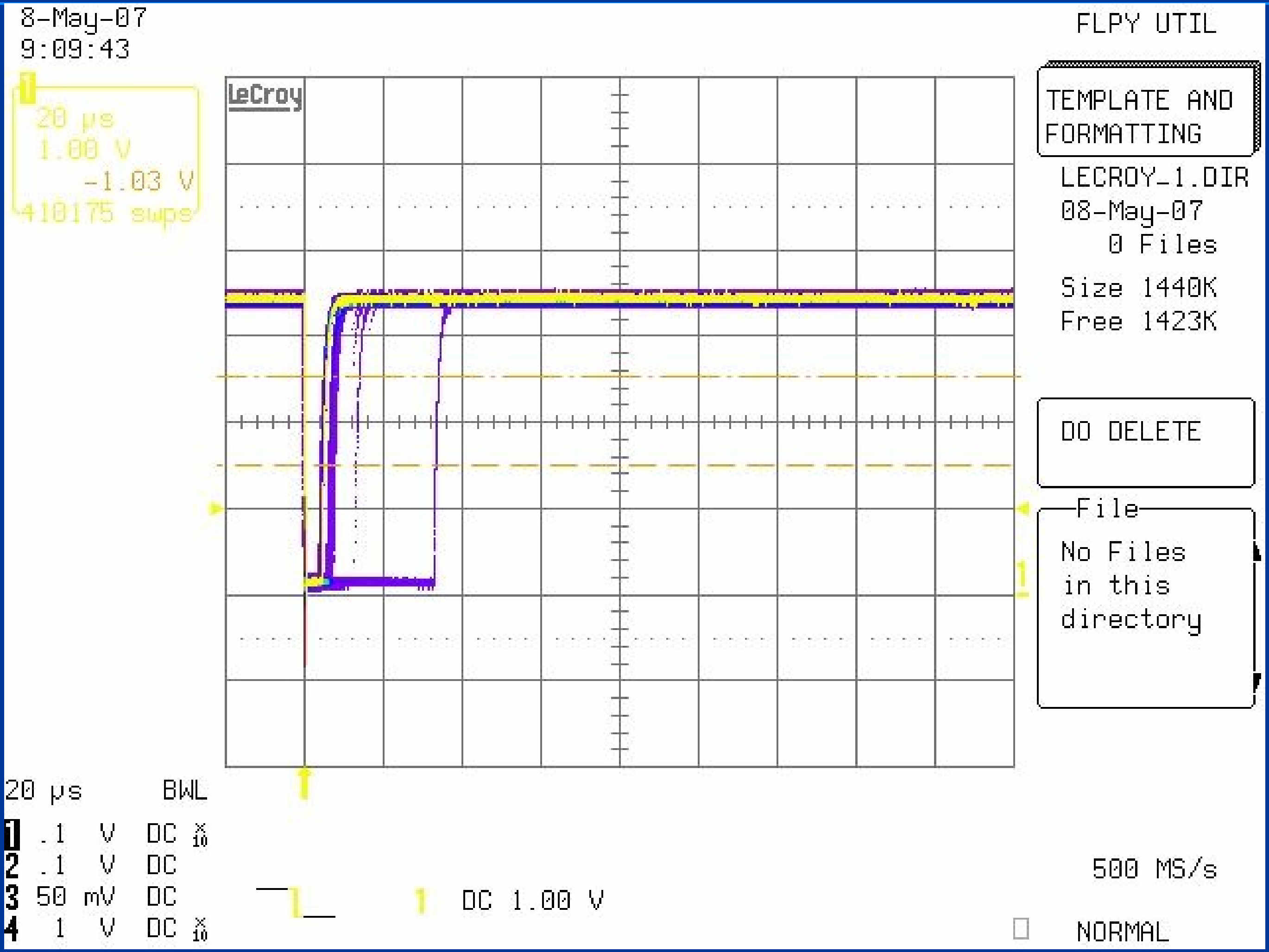
More Sharp Tools

- PCI bus analyzers - how do you prove it's working / not-working?
- OPB-PCI bridge - no known device driver, have to build one
- Early development - device drivers become part of Verification & Validation



GENERAL DYNAMICS
Canada

Logic Analyzers



```
8-May-07
9:09:43
```

```
20 µs
1.00 V
   -1.03 V
410175 swps
```

```
LECROY_1.DIR
08-May-07
   0 Files
Size 1440K
Free 1423K
```

```
No Files
in this
directory
```

```
20 µs        BWL
1  .1   V   DC x10
2  .1   V   DC
3  50  mV   DC
4   1   V   DC x10
```

```
DC 1.00 V
500 MS/s
NORMAL
```

GENERAL DYNAMICS
Canada

Hardware assisted debugging

- You need it, you just don’t know it yet…
  - Squirt kernel’s onto the board prior to your boot code being available
  - Your boot code will be late, and buggy and late…
- Symbolic debugging of your kernel as it comes up is just so ….. decadent

Abatron – Don’t leave the design stage without it!

Awesome Things

- Device Independence is a good/great/AWESOME thing
- BUS-Device abstraction is a good/great/AWESOME thing
- NetBSD’s device independent nature is both a blessing and a curse.

Cross Development

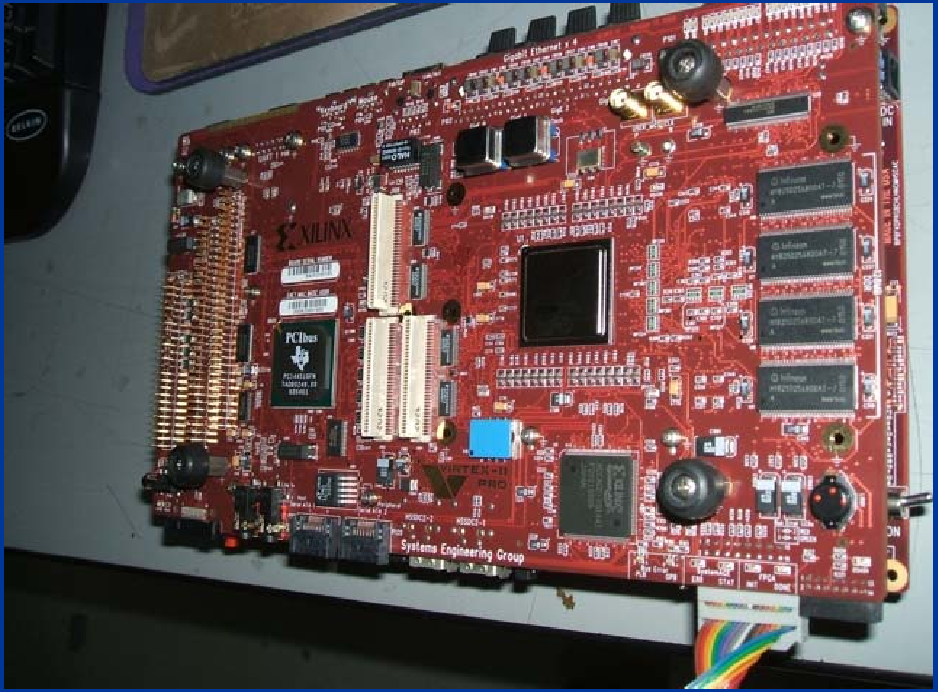
- Cross development can save your bacon
- You must admit to yourself upfront that cross development is harder, and takes more awareness and effort to reap the benefits
- Not everyone will make this conceptual leap

Cross Development – Messy Details

- Proveout on x86 PCs
  - -Target Development Board (Generally commercially available) ML-300/ML-403
- Target hardware will be late (and buggy)
  - (Customer Specific - restricted access)
- Done right, take the CF card with FPGA Bitstream from development board and plug CF into target device and it should just work
- If byte sex scares you, give up now

Cross Development – Target Hardware

- Make sure it's close to what you need to deliver
- PowerPC 40x CPU
- PCI expandability
- Boots from CF card

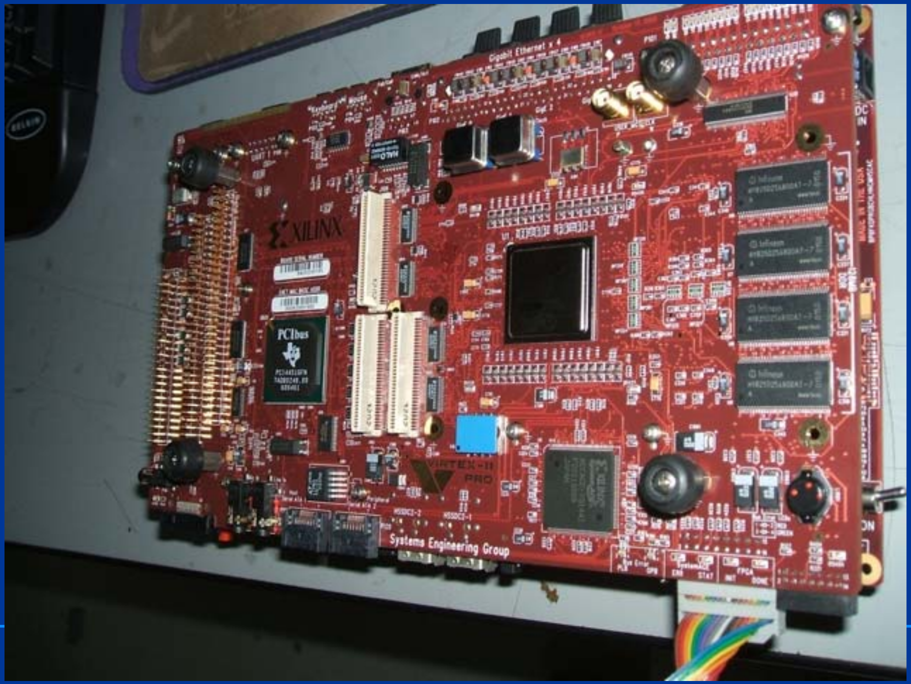


Cross Development – new direction

- Faster processors for less money are always welcome

Cross Development - Example

- Start with commercial development board
  - Xilinx ML-300/Xilinx ML-403
  - Multiple cores == Multiple CPUs, *Tasty!!!*



Cross Development – Commercial Support

- Can you boot and run your chosen Operating System on the target development board?
- One neck to choke when it doesn’t work
- FPGA allows one to make the final target board look and behave -*EXACTLY*- like your development board, from a memory layout perspective.

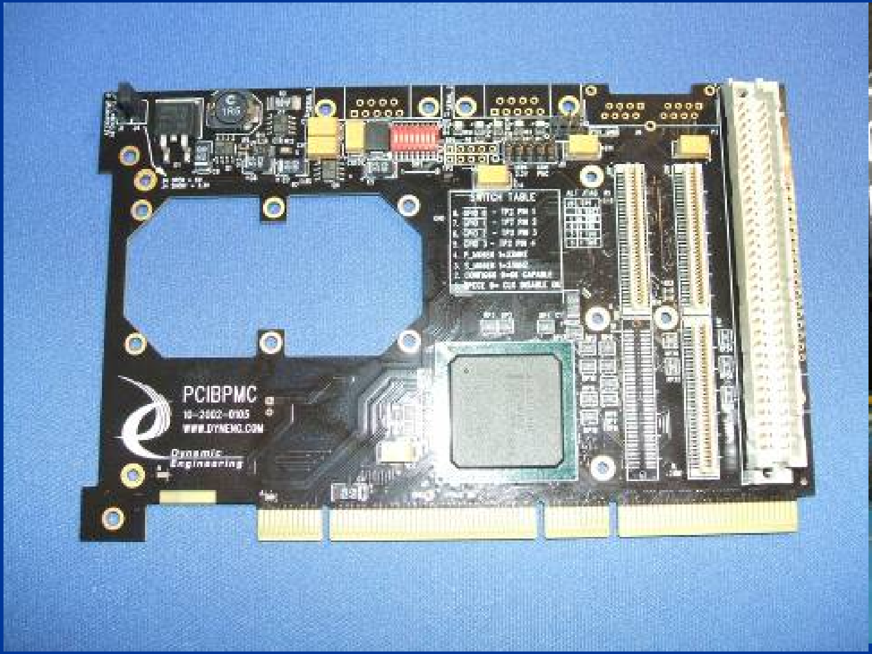
Cross Development – Big Win

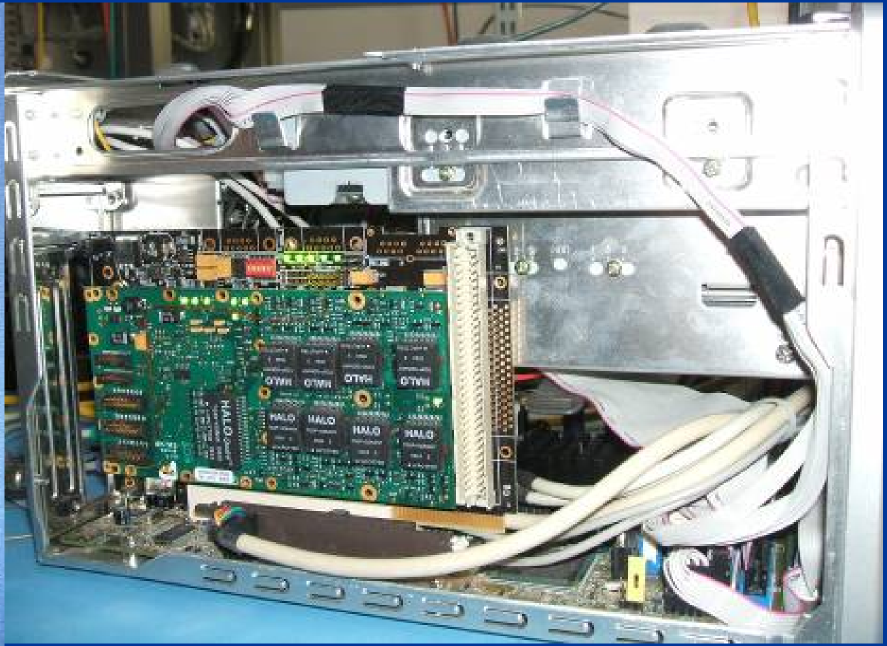
- Example:
  - Target peripheral board for T1/E1
  - Final processor board not yet available
  - Doesn't have an Operating System on it yet?

Cross Development – Game Plan

- No problem - can target hardware run in a PC?
- Yes? The addition of a PCI-PCI adaptor, allowed us to develop our driver on a PC, and then cross compile it for the target device whence it was ready
- People will think *You Insane* for proposing this

PCI Bus Adapters





GENERAL DYNAMICS
Canada

Cross Development – Keeping your boss's boss OFF your boss's back

- Managerial types then get to schedule SW development efforts which overlap with HW development efforts and look like managerial geniuses.
- Allowing efforts to be parallel-izable is a good thing.

Then you have a chance of success !!

Then you have to answer the question.. Why didn't we always do it like this ??

Embedding

- FPGA's make for easier development of "embeddable" hardware.
- What does it mean to be embeddable?

Embedding Summary

- Embedding - Nothing that moves
- It moves – It DIES
    - IMO F. Gomez

Embedding - Nothing that moves

- The idea's from nanoBSD definitely apply
- PHK's Law
  - Nothing that rotates!
- Restating PHK's Law:
  - If it goes round, it's gonna go down

Failures in Embedded Systems

- If it goes down;
  - angry customer's with firearms and intimate knowledge of physical violence are really bad news

GENERAL DYNAMICS
Canada

Extremes

- Extreme's of temperature, humidity and vibration
  - All of which are completely deadly to normal physical things that move
- No hard drive – no rotating magnetic media
- Cooling will be an issue, especially with no fans
- Vibration will reduce all of your carefully SMT placed, judiciously re-flowed components to trailmix faster than you think

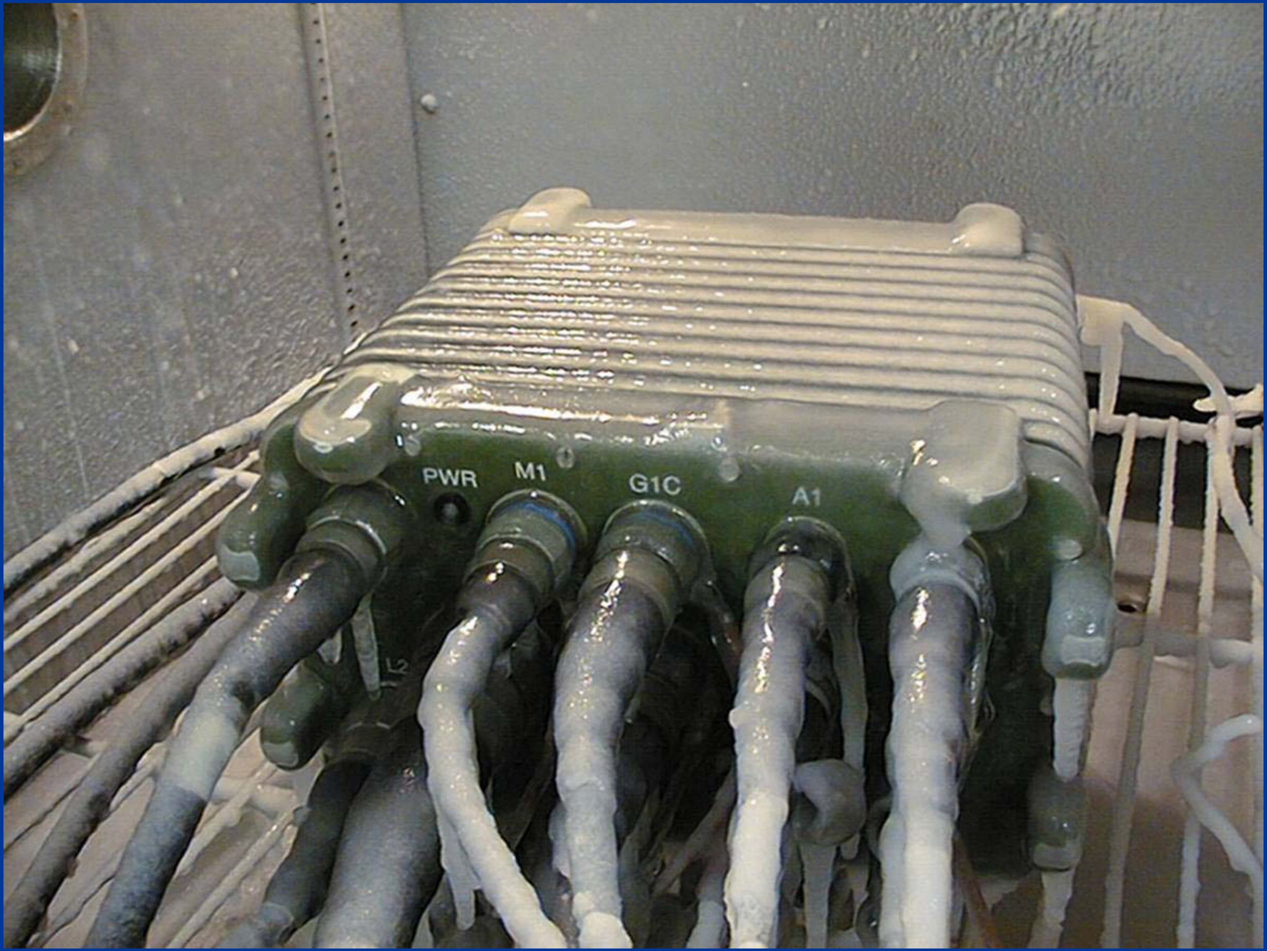
11:45 AM in Basra, Iraq

Humidity Extremes

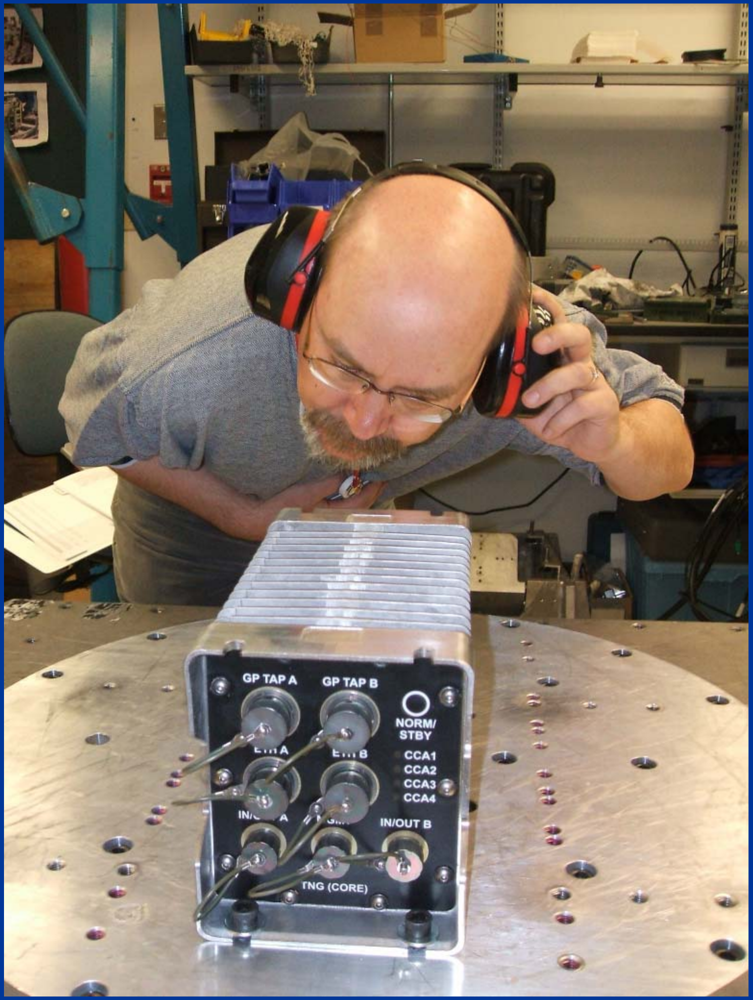
GENERAL DYNAMICS
Canada

Temperature Extremes

Combination of Temperature & Humidity



Vibration

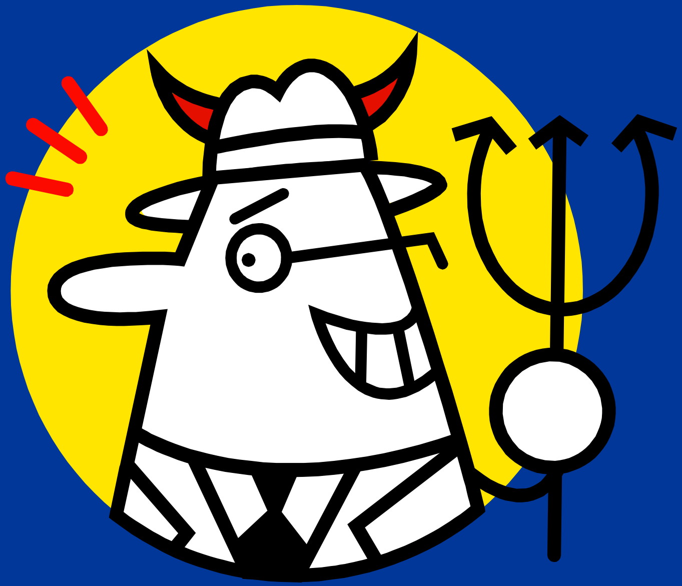

GP TAP A
GP TAP B
NORM/
STBY
ETH A
ETH B
CCA1
CCA2
CCA3
CCA4
IN/OUT B
TNG (CORE)

Physics is not necessarily your friend

- Nothing says “I don’t care” like blunt force impact trauma

Chip Level Flash Must DIE

- Soldered on chip level flash is EVIL!



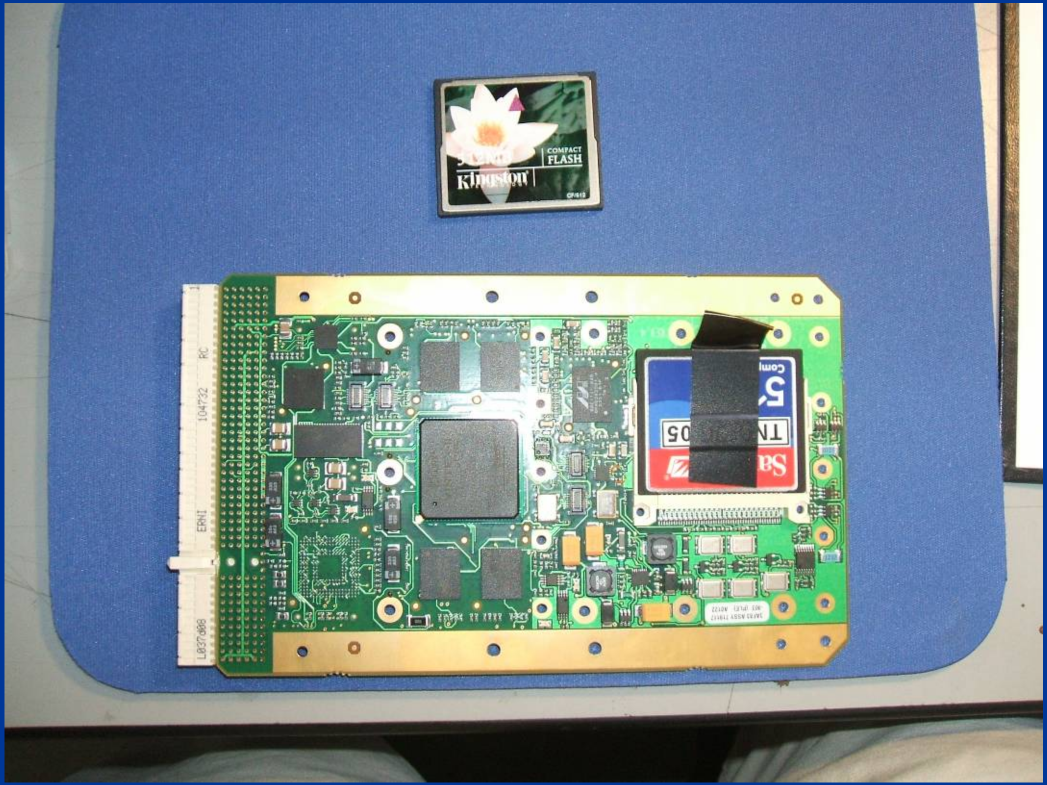
GENERAL DYNAMICS
Canada

Compact Flash is your FRIEND

- So we use industrial spec. Compact Flash
- CF cards fail vibration test, so we glue them into place [ARRRRGGHHHH!!!!]
- CF capacities are expanding and I don't know of any software project that got smaller across time.....

Compact Flash

"Patented Universal Removal Tab"



Embedded Projects..and YOU

- Things that matter
- Things that don't
- What's the difference

Things that Matter!!

- The class of things that matter is infinite
- The class of things that don’t matter is negligible
- The important thing is PRIORITY

GENERAL DYNAMICS
Canada

Mistakes

- Make sure you learn from the mistakes of others because you surely don’t have time to make them all yourself
- You have the capability to learn from your mistakes – you will learn a great deal today
- Make sure to make new mistakes
- Make it hard for other people to make your mistakes

Random Events you hadn’t thought about

- Things will break that you did not expect to break
  - What does “broken half-word multiply” mean?
  - Radioactive decay, why should I care?
  - What really matters is your response

Device Drivers and Boot Code

- Why are they so hard to write?
- Why are they so hard to debug?
- Why is it so hard to find people who understand that they are hard to write and hard to debug?
- Conventional software estimating tools get “divide-by-zero” errors when dealing with device driver code
- Hardware folks perspective of “easy” is somewhat different that software “easy”

Embedded Systems – Rules of Thumb

- It's not paranoia if they really are out to get you……
  - Everyone and Everything is going to conspire to make your job difficult, so take every possible opportunity presented to try and make things easier
  - Checksums for stuff that transits a suspected dodgy interface
  - Verifying that a signal stays steady, etc…

Miss and Kiss

- Make it Simple Stupid
- Keep it Simple Stupid
- Anyone can make it complex, your job is to make it simple

- DON’T BUILD IT IF YOU CAN’T TEST IT

Embedded Systems – Getting Easier?

- If it was easy, everyone would do it!

  -- Tom Hanks – A league of their own

- *FPGAs are a big step toward making it easier*