# synthesizing Forth

Forth in the satellite industry

a report on work in progress

Klaus.Schleisiek at spacetech-i.com

## Design iterations

A test cycle in VHDL (modification, re-synthesis, place&route, FPGA re-configuration) takes about 45 minutes.

Putting uCore into the FPGA, realizing "higher level" functionality in software cuts a test cycle down to 15 seconds.

But:

- uCore is not qualified for space use
- Software qualification is much more expensive than VHDL qualification, because we do not quite know "how" to do that ;)

=> Software can only be used during the BB and EM phases :(

## Peculiarities of the space industry

Usually these are one-off projects.

Once deployed, satellites can not be repaired.

Development is usually done in four stages:

- BB: Breadboard (get the functionality right)
- EM: Engineering model (eventual form and function)
- QM: Qualification model (heavily mistreated using qualified parts)
- FM: Flight model (clean room assembled)

## Consequences

Forth programming is restricted to design exploration.

- For the QM and FM phases, uCore and Forth have to be "designed out" and replaced by VHDL code.

=> Forth should be written with synthesizability in mind.

How do we have to write Forth to make it easily portable to VHDL?

## Qualified parts

These are the most important criteria:

- Shock and vibration (during satellite launch)
- Extreme temperature range
- Radiation tolerance depending on the intended orbit:
  - Total dose (blurring the IC structures created by diffusion)
  - SEE: Single event effects due to heavy ions
  - Latchup due to heavy ions

## Forth / VHDL

Forth is a sequential language.

Parallelism must be mimicked using multi-tasking.

VHDL describes parallel processes, which all happen "at the same time". Sequential behaviour must be explicitly designed in if needed.

=> Forth should be written in a data flow style to ease the process of porting it to equivalent VHDL code.

## FPGA design flow

Simplified ESA design flow for re-programmable FPGAs:

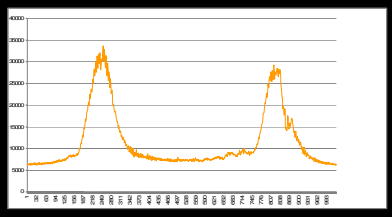
- Definition phase
- Top-down architectural design
- PDR: Preliminary Design Review
- Bottom-up RTL code creation
- VHDL simulation and design verification
- FPGA synthesis and place&route
- Design validation on EM hardware
- CDR: Critical Design Review
- Design for radiation tolerance
- QR / AR: Quality / Acceptance Review

Each warning generated by the design tools has to be explained to beaurocrats in detail.

## An example

1064 nm laser wavemeter:

An InGaAs line of 1024 pixels must be read out, low-pass filtered (noise suppression) and differentiated to find the center of a gaussian shaped interferometer "fringe" within 300 usec. Readout alone takes 200 usec.



# 1st Forth approach

The 1024 AD-converted pixel values will be stored at SCAN in Forth's data memory by a state machine.

```
: initial  ( -- )
   0   Scan 1-
   Edge @ ?FOR  1 + ld >r - r>  NEXT  Crest @ +
   Edge @ ?FOR  1 + ld >r + r>  NEXT  drop
   first-pixel diff!
;
: filter ( i -- i+1 )
   dup >r   dup diff@ >r   first-pixel -
   Scan + ld   Edge @ + ld   Crest @ + ld   Edge @ + @
   -rot + - + r> +   r> 1+   tuck diff!
;
: differentiate  ( -- )
   Diff #pixels erase   initial
   first-pixel  BEGIN  filter   dup last-pixel = UNTIL
   drop
;
```

# synthesizable approach

```
Variable Lead

: filter  ( sum I -- sum' )   Lead @
   IF  dup Edge @             u< IF  pix@ -  EXIT THEN
      Edge @ Crest @ + 1- over u< IF  pix@ +  EXIT THEN
      drop  EXIT
   THEN
   Scan + 1- ld   Edge @ - ld   Crest @ - ld   Edge @ - @
   -rot + - + +
;
: differentiate   ( -- )
   Diff #pixels erase   Lead on   0   #pixels 0
   DO  I filter   I window = IF  Lead off  THEN
       Lead @ 0= IF  dup I first-pixel - diff!  THEN
   LOOP  drop
;
```