noForth voor RP2350

0) Waarom voor de RP2350

Hebben we niet genoeg aan de RP2040? Jawel vaak wel, maar zoals Jeroen liet zien is de RP2350 is een heel veelzijdige chipje.

1) Wat is er nodig

We moeten een Pico-bin image-definition toevoegen om de binary te duiden, de secundairy boot herschrijven. Nieuwe assembler en disassembler en compleet herschreven boot routines, etc.

FFFFDED3	Magic header number
10210142	Secure ARM image type
000001FF	Size & last item
00000000	Has only a single block
AB123579	Magic footer number

2) Het aanpakken van noForth t

Eerst het simpele knipperlichtje omgezet in noForth assembler.
Wat moest er gedaan worden:

- De noForth t (dis)assembler uitbreiden met M33 opcodes
- De Image-definition of pico-blok toevoegen
- De UF2 generator aanpassen (een kleinigheidje)
- De noForth t sources volledig doorvlooien, adressen, constanten, offsets en code aanpassen
- De nieuwe co-processor opcodes begrijpen en testen
- RS232 erbij en testen
- Stap voor stap de veranderde boot code toevoegen en testen, dit heb is van achter naar voren gedaan
- Assembler code, met M33 opcodes aangepast
- Exception handler is veelzijdiger (Bus fout/Mem. fout)/Etc.)

3) noForth t solo bouwen

De gehele source doorgelopen om de gewijzigde register adressen en code aan te passen. B.V. het aanzetten van de klok generators. Vele routines krijgen kleine aanpassingen andere zijn herschreven. De hardware vermenigvuldiger is verwijderd en vervangen door de UMULL en SMULL opcode. /MOD MOD en / worden nu met de SDIV en de MLS opcode gebouwd. Voorbeeld:

```
forth: code /MOD    ( x y -- mod q )
    day  sp ) ldr,         \ DAY = x, TOS = y
    sun day tos sdiv,      \ Signed 32/32 divide, SUN = q
    day sun tos day mls,   \ Multiply & subtract: mod = x – (q * y)
    tos sun movs,          \ DAY = DAY - (SUN * TOS) so DAY is remainder
    day  sp ) str,
    next,
end-code
```

- Als eerste de met het knipperlichtje geteste opstart code gebruikt, om de eerste bugs in het verdere opstart proces op te sporen.

- Om de MS routine te laten werken moet één van de zes TICKS generators starten. Met timer0 die hierdoor aangestuurd wordt bouwen we MS en US.

- Het volgende lastige punt is het gebruik van de ROM API's vanwege de beveiliging in de RP2350, moeten deze op een andere manier aanroepen worden. Dat had flink wat voeten in aarde, omdat de C-routine die als documentatie dient maar moeilijk doorgrond kon worden. Ulli's Claude AI gaf ons een hint.

```
inside: routine ROMCALL ( -- a ) \ In:  DAY=pointer to lookup string
    { ip sp w tos lr } push, \ Save data for real function call
    ip  day ) ldrh,         \ IP = R0 = lookup code
    sp 4 # movs,            \ SP = R1 = ARM secure key, 0A = Nonsecure!
    w 0 # movs,             \  W = R2 = ROM base address
    hop  w 16 #) ldrh,      \ HOP = address of lookup subroutine
    hop blx,                \ Call lookup table
    sun ip mov,             \ Copy subr. address to SUN
    { ip sp w tos pc } pop, \ Restore data for function call
end-code
```

Nu dit alles functioneert, kon de solo versie verder getest en gedebugged worden. Al snel kon daar de simpele USB zonder multitasker aan toegevoegd worden. Die functioneert na twee kleine aanpassingen, één bit-masker en één register adres.

4) Vervolg stappen

noForth t solo met multitasker is daarna aan de beurt. Het meeste werk is de controle van de systeemvariabelen. De structuur daarvan is voor de multitasker versie anders. De USB source heeft naast het bit masker en register adres nog één extra aanpassing nodig. Al weet ik nog niet waarom...
Dan de duo versie zonder multitasker. Daar zit vooral de IO co-processor in de weg voor een vlot resultaat. De coprocessor moet voor elke CPU core apart aangezet worden. De geldt overigens voor alle aanwezige coprocessors. Weer een lesje geleerd.

```
\ Enable coprocessor P0
    hop  E000ED88 #w mov32,         \ CPACR  Enable access to copocessors
    day  hop ) ldr,
    w 3 # movs,                     \ CP0 full access, bit 0 & 1 high
    day w orrs,
    day  hop ) str,
    dsb,                            \ Mandatory opcodes
    isb,
```

Tenslotte de duo versie met multitasker. De meest complexe. Die stap was lastig, omdat ik me vergiste in een tabel. En wel een tabel die is opgenomen in een stukje initialisatie code.
De opstart code crashte daardoor al in een zeer vroeg stadium.
Het gevolg hiervan is dat de noForth t sources nu zeer netjes zijn. Dit dankzij vele kleine verbeteringen, die gemaakt zijn omdat deze laatste noForth niet op wilde starten. De meeste voorbeeld code draait al weer.

5) Finale

- noForth moet nu goed aan de tand gevoeld worden
- De bibliotheek is bijna geheel gecontroleerd en moet afgerond en getest worden
- Het nieuwe klok systeem dat Jeroen heeft bedacht, moet toegevoegd en aan de tand gevoeld worden
- Snelle bit operatoren aanbrengen, dat vergt nog een ontwerp beslissing
- Een floating point bibliotheek schrijven en toevoegen

6) Nog vragen?