HOGESCHOOL ROTTERDAM

Real-Time Operating Systems

ROS01

Minor Embedded Systems

Week 2
Super loop construct with an ISR

versd@hr.nl
brojz@hr.nl

Planning ROS01

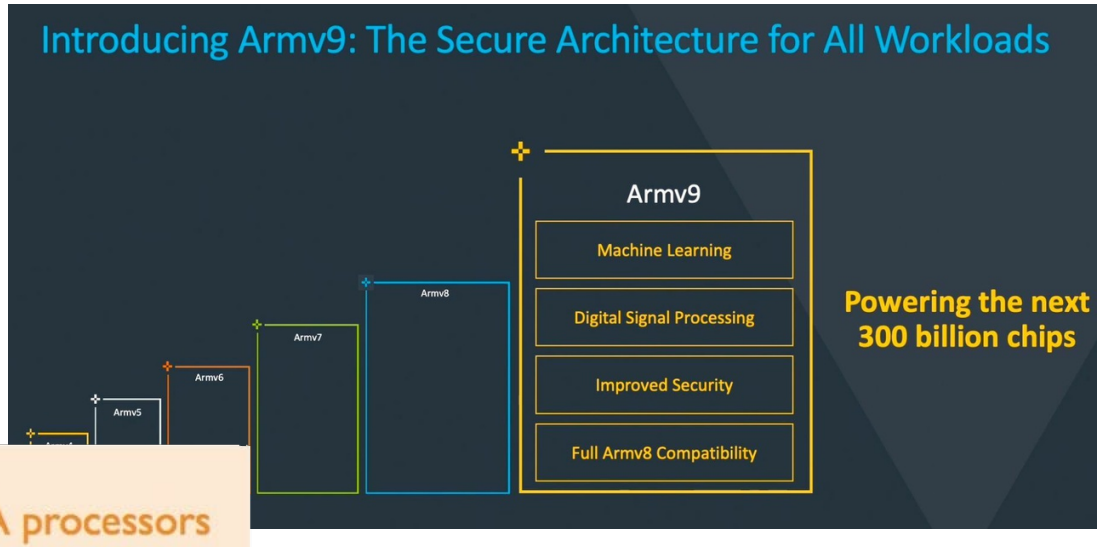
- Week 1: Introduction – Hello World
- **Week 2: Super loop construct with an ISR**
- Week 3: Cooperative Scheduling
- Week 4: Pre-emptive Scheduling
- Week 5: Using TI-RTOS
- Week 6: Schedulability Analyses, Priority Assignment
- Week 7: Response Time Analyses
- Week 8: Finalizing Final Assigment

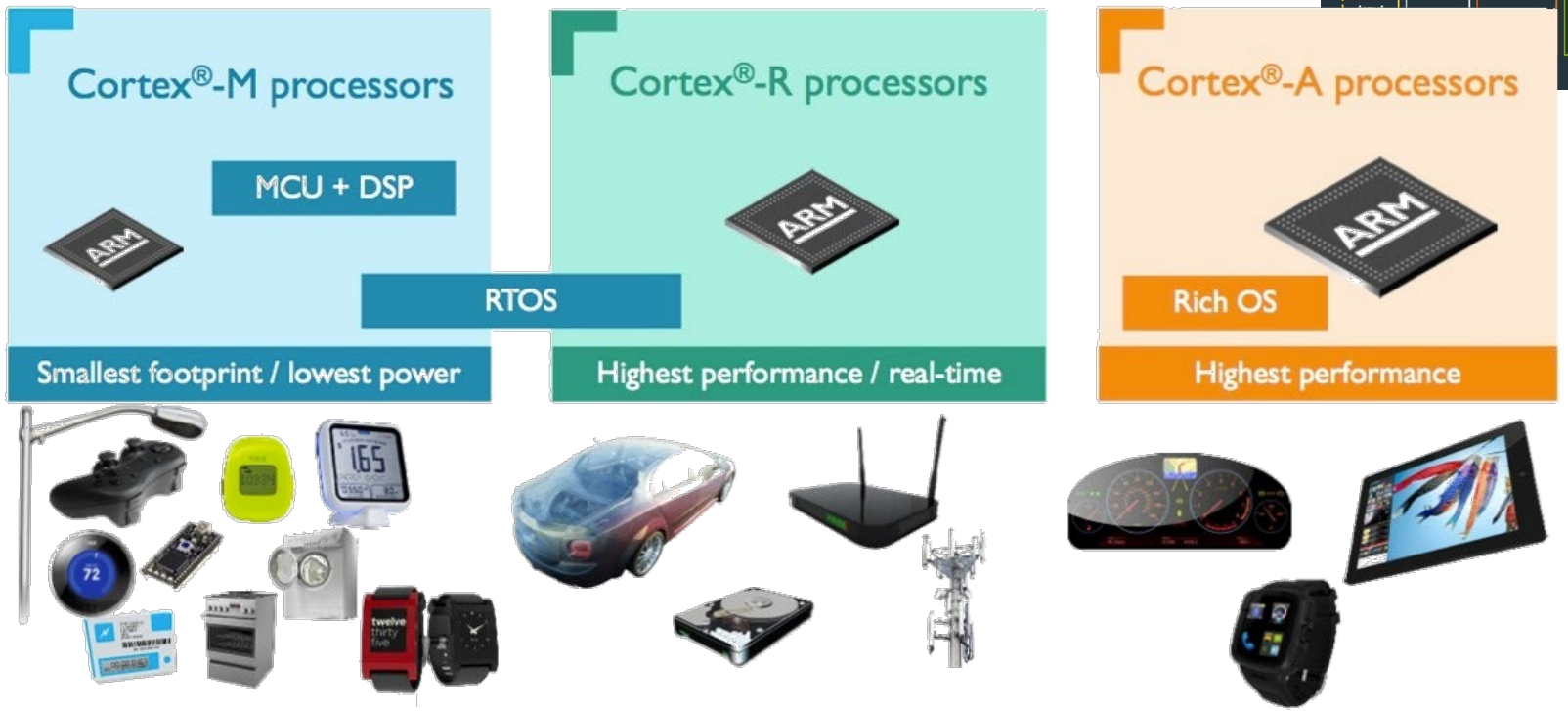
Overview

- The microcontroller
  - History
  - Properties of the Cortex M4 CPU
  - Interrupt system
  - Timers
  - Direct Memory Access system
- Superloop ‘scheduler’

ARM History

- Started in 1980 as a co-processor
  - ARM1 architecture
- Latest:
  - ARMv9-A architecture (zie bijv. Cortex-A710)



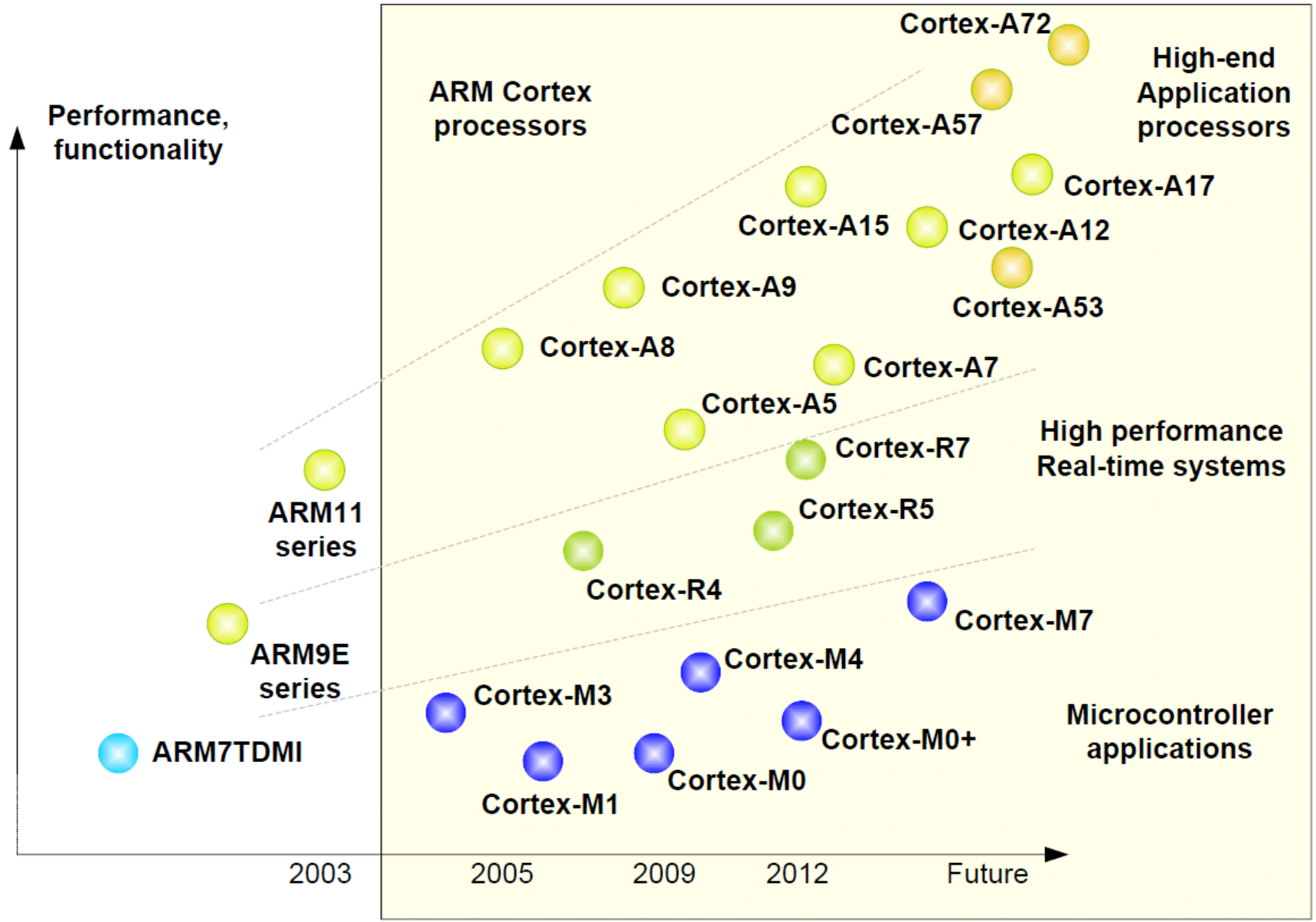


Difference to x86?

exceed expectations

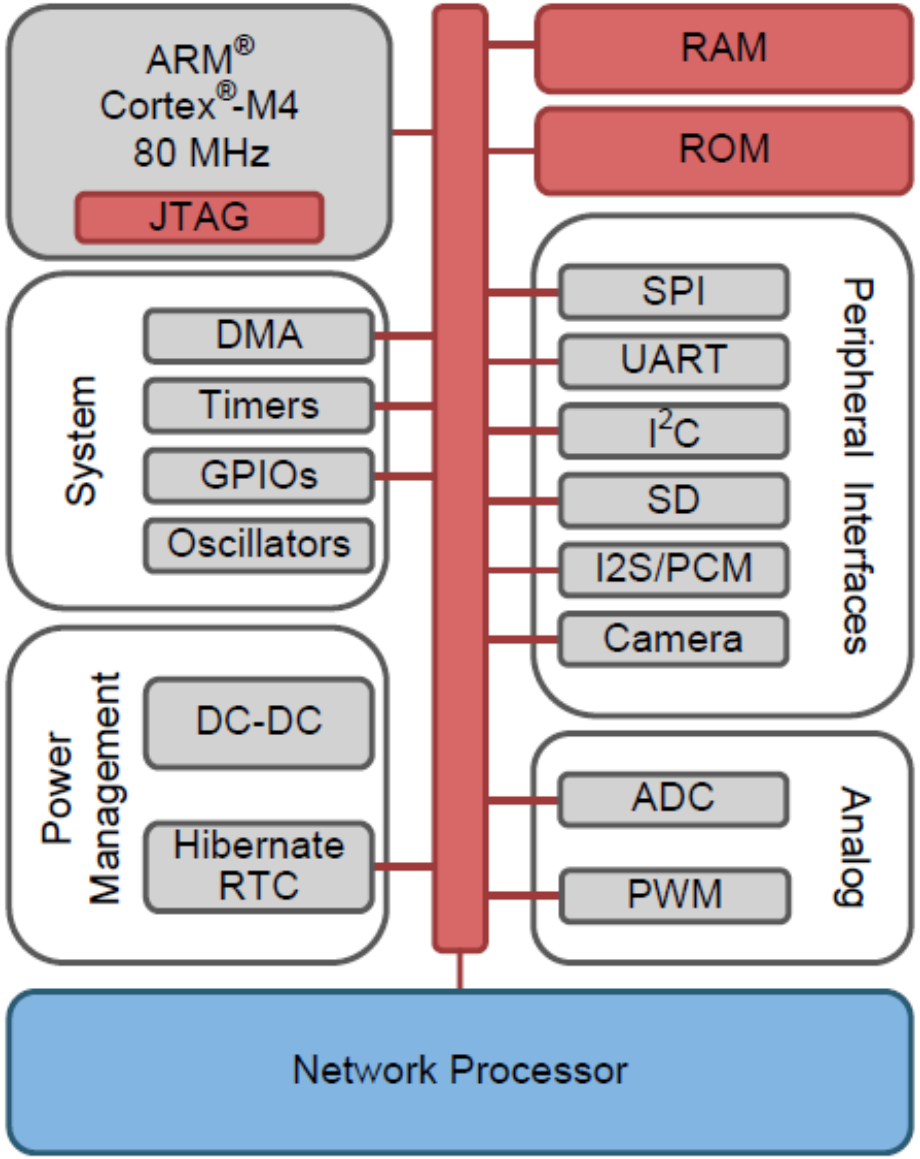
HOGESCHOOL ROTTERDAM

ARM family



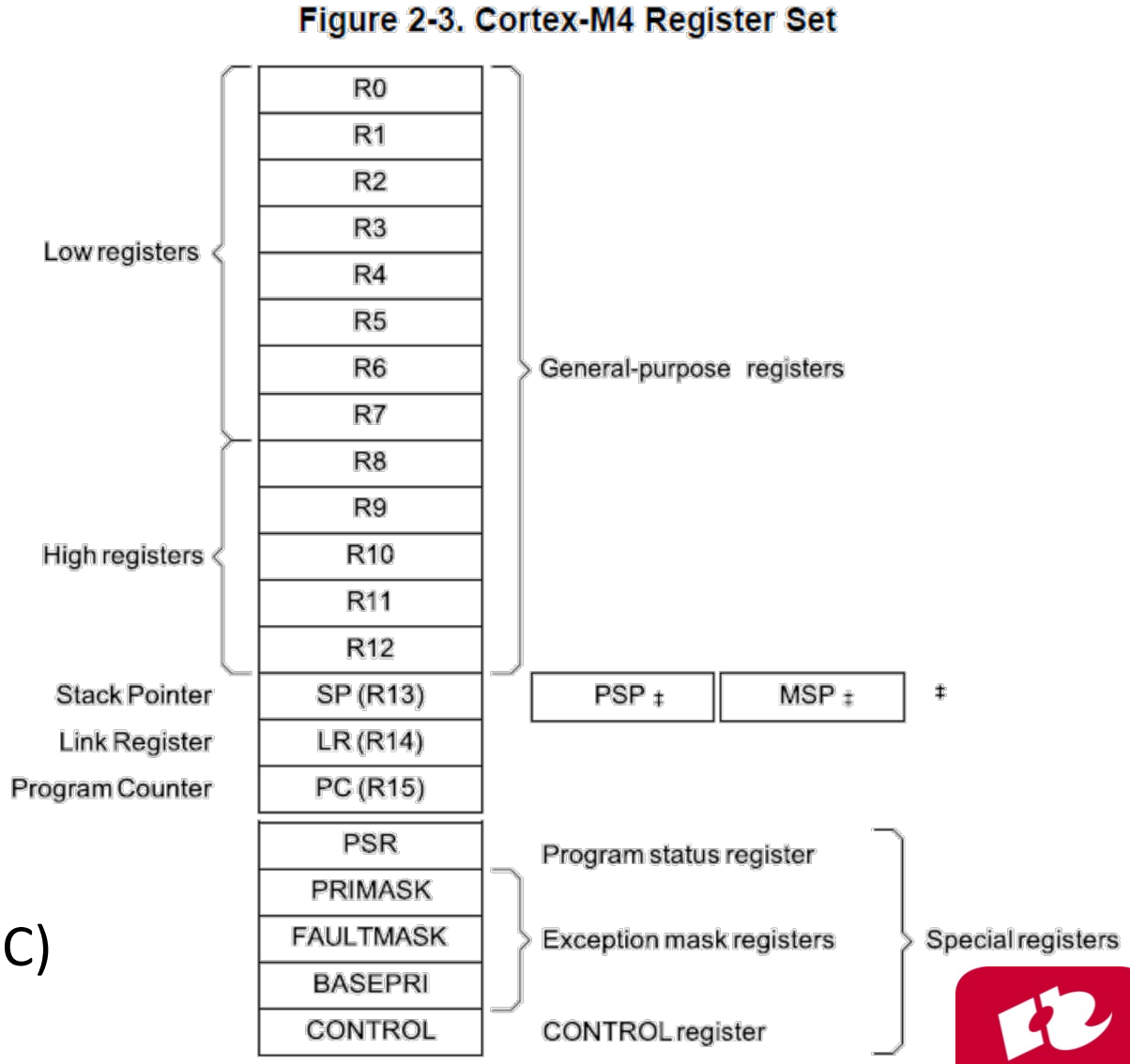
HOGESCHOOL ROTTERDAM

CC3220S



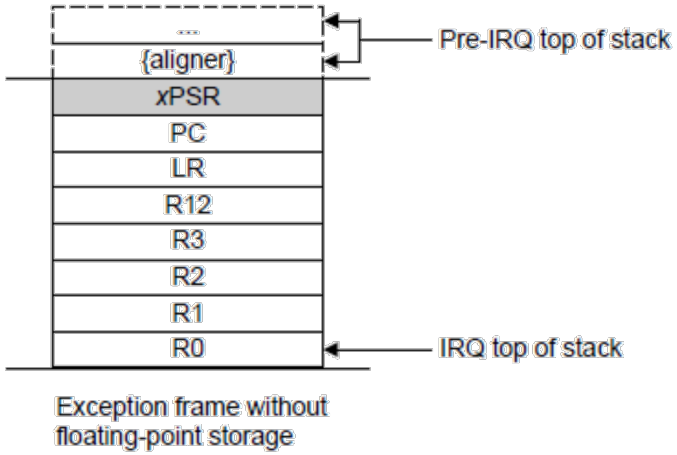
Cortex M4

- ARMv7E-M architecture
  - Two modes of operation: Thread (application code) and Handler (exceptions and OS).
    - Main SP(MSP) / Process SP (PSP)
  - Two privilege levels: Unprivileged and privileged.
  - Thumb 2 instructions
    - Single cycle integer division/multiply
  - Ultra-low power sleep modes
  - Atomic bit manipulation (Bit-banding)
  - System timer (SysTick)
  - Nested Vector Interrupt Controller (NVIC)



Exceptions – NVIC

- Nested Vectored Interrupt Controller
  - 75 exception sources
  - Priority level between -3 to 7 (low number is high priority)
    - Priority groups
      - group priority (determines preemption)
      - sub priority fields (determines order when multiple interrupts are pending)
  - Tail-chaining of Exception Handlers
  - External Non Maskable Interrupt



Exception types

Exception Type	Vector Number	Priority[a]	Vector Address or Offset[b]	Activation
-	0	-	0x0000.0000	Stack top is loaded from the first entry of the vector table on reset.
Reset	1	-3 (highest)	0x0000.0004	Asynchronous
Non-Maskable Interrupt (NMI)	2	-2	0x0000.0008	Asynchronous
Hard Fault	3	-1	0x0000.000C	-
Memory Management	4	programmable[c]	0x0000.0010	Synchronous
Bus Fault	5	programmable[c]	0x0000.0014	Synchronous when precise and asynchronous when imprecise
Usage Fault	6	programmable[c]	0x0000.0018	Synchronous
-	7-10	-	-	Reserved
SVCall	11	programmable[c]	0x0000.002C	Synchronous
Debug Monitor	12	programmable[c]	0x0000.0030	Synchronous
-	13	-	-	Reserved
PendSV	14	programmable[c]	0x0000.0038	Asynchronous
SysTick	15	programmable[c]	0x0000.003C	Asynchronous
Interrupts	16 and above	programmable[d]	0x0000.0040 and above	Asynchronous

Table 2-11. Faults

Fault	Handler	Fault Status Register	Bit Name
Bus error on a vector read	Hard fault	**Hard Fault Status (HFAULTSTAT)**	VECT
Fault escalated to a hard fault	Hard fault	**Hard Fault Status (HFAULTSTAT)**	FORCED
MPU or default memory mismatch on instruction access	Memory management fault	**Memory Management Fault Status (MFAULTSTAT)**	IERR [a]
MPU or default memory mismatch on data access	Memory management fault	**Memory Management Fault Status (MFAULTSTAT)**	DERR
MPU or default memory mismatch on exception stacking	Memory management fault	**Memory Management Fault Status (MFAULTSTAT)**	MSTKE
MPU or default memory mismatch on exception unstacking	Memory management fault	**Memory Management Fault Status (MFAULTSTAT)**	MUSTKE
MPU or default memory mismatch during lazy floating-point state preservation	Memory management fault	**Memory Management Fault Status (MFAULTSTAT)**	MLSPERR
Bus error during exception stacking	Bus fault	**Bus Fault Status (BFAULTSTAT)**	BSTKE
Bus error during exception unstacking	Bus fault	**Bus Fault Status (BFAULTSTAT)**	BUSTKE
Bus error during instruction prefetch	Bus fault	**Bus Fault Status (BFAULTSTAT)**	IBUS
Bus error during lazy floating-point state preservation	Bus fault	**Bus Fault Status (BFAULTSTAT)**	BLSPE
Precise data bus error	Bus fault	**Bus Fault Status (BFAULTSTAT)**	PRECISE
Imprecise data bus error	Bus fault	**Bus Fault Status (BFAULTSTAT)**	IMPRE
Attempt to access a coprocessor	Usage fault	**Usage Fault Status (UFAULTSTAT)**	NOCP
Undefined instruction	Usage fault	**Usage Fault Status (UFAULTSTAT)**	UNDEF
Attempt to enter an invalid instruction set state [b]	Usage fault	**Usage Fault Status (UFAULTSTAT)**	INVSTAT
Invalid EXC_RETURN value	Usage fault	**Usage Fault Status (UFAULTSTAT)**	INVPC
Illegal unaligned load or store	Usage fault	**Usage Fault Status (UFAULTSTAT)**	UNALIGN
Divide by 0	Usage fault	**Usage Fault Status (UFAULTSTAT)**	DIV0

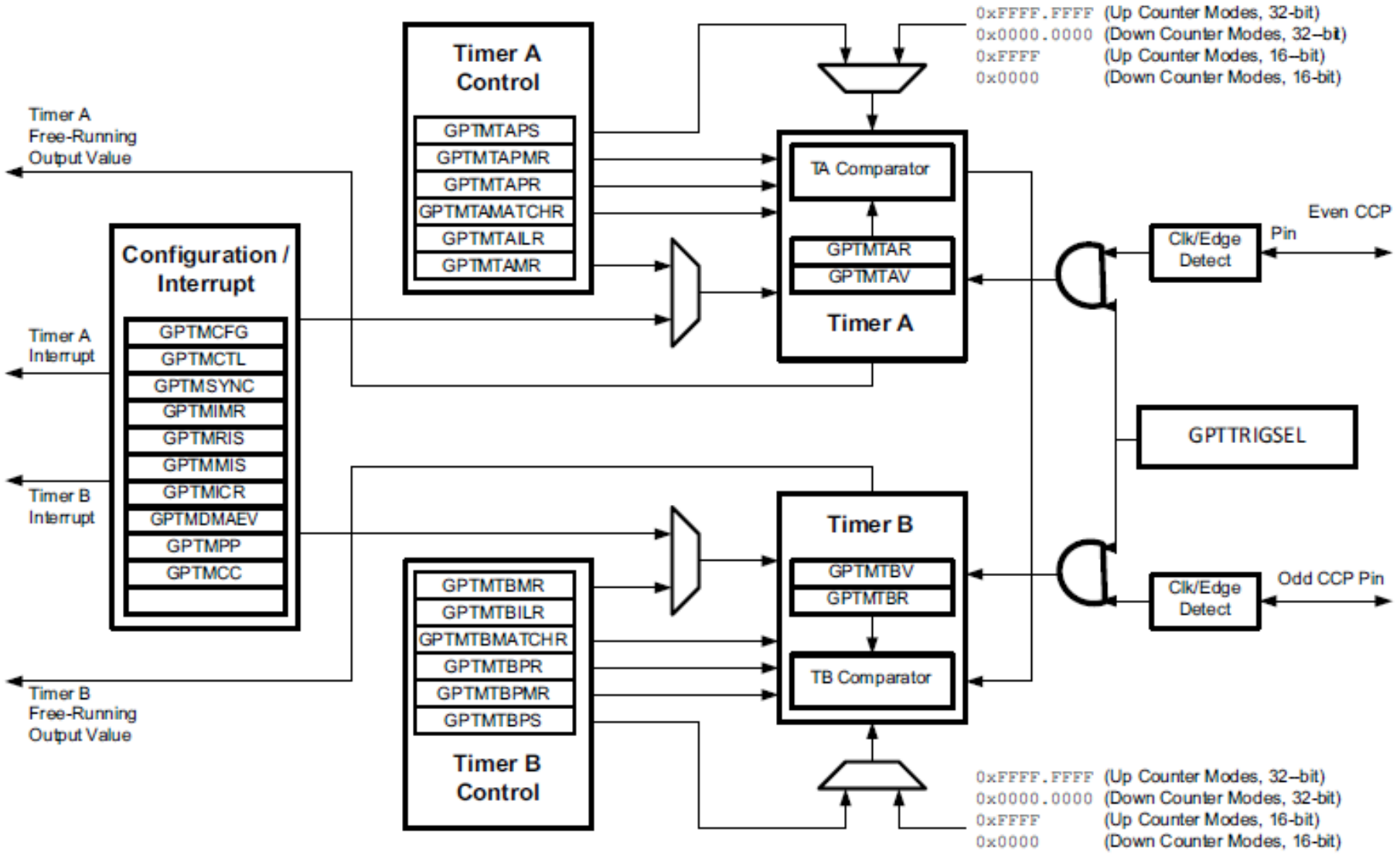
Timers

- 4 GPTM (16/32)
  - two 16 bit counters per module
  - OR one 32 bit counter per module
    - How long does an overflow take?

2^{32} ≈ 4.3e9 ≈ 53.7 s @ 80MHz

Timers

Figure 9-1. GPTM Module Block Diagram



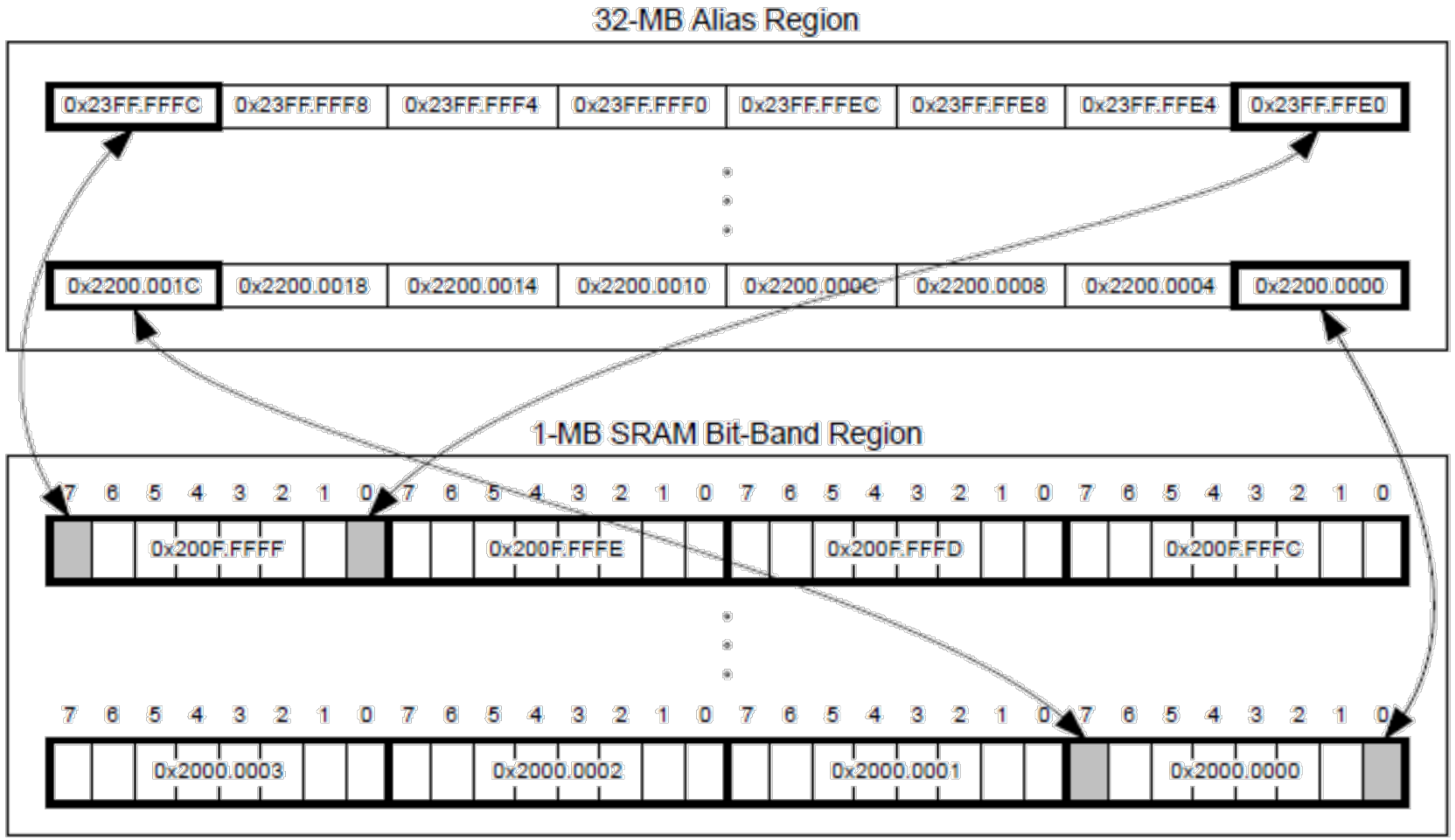
Bit-Banding

Table 2-5. SRAM Memory Bit-Banding Regions

Address Range			
Start	**End**	**Memory Region**	**Instruction and Data Accesses**
0x2000.0000	0x2003.FFFF	SRAM bit-band region	Direct accesses to this memory range behave as SRAM memory accesses, but this region is also bit addressable through bit-band alias.
0x2200.0000	0x23FF.FFFF	SRAM bit-band alias	Data accesses to this region are remapped to bit band region. A write operation is performed as read-modify-write. Instruction accesses are not remapped.

Bit-banding for peripherals is not supported in the CC32xx.

Bit-Banding



You should use bit0 in the alias region to change a bit in the bit-band region.

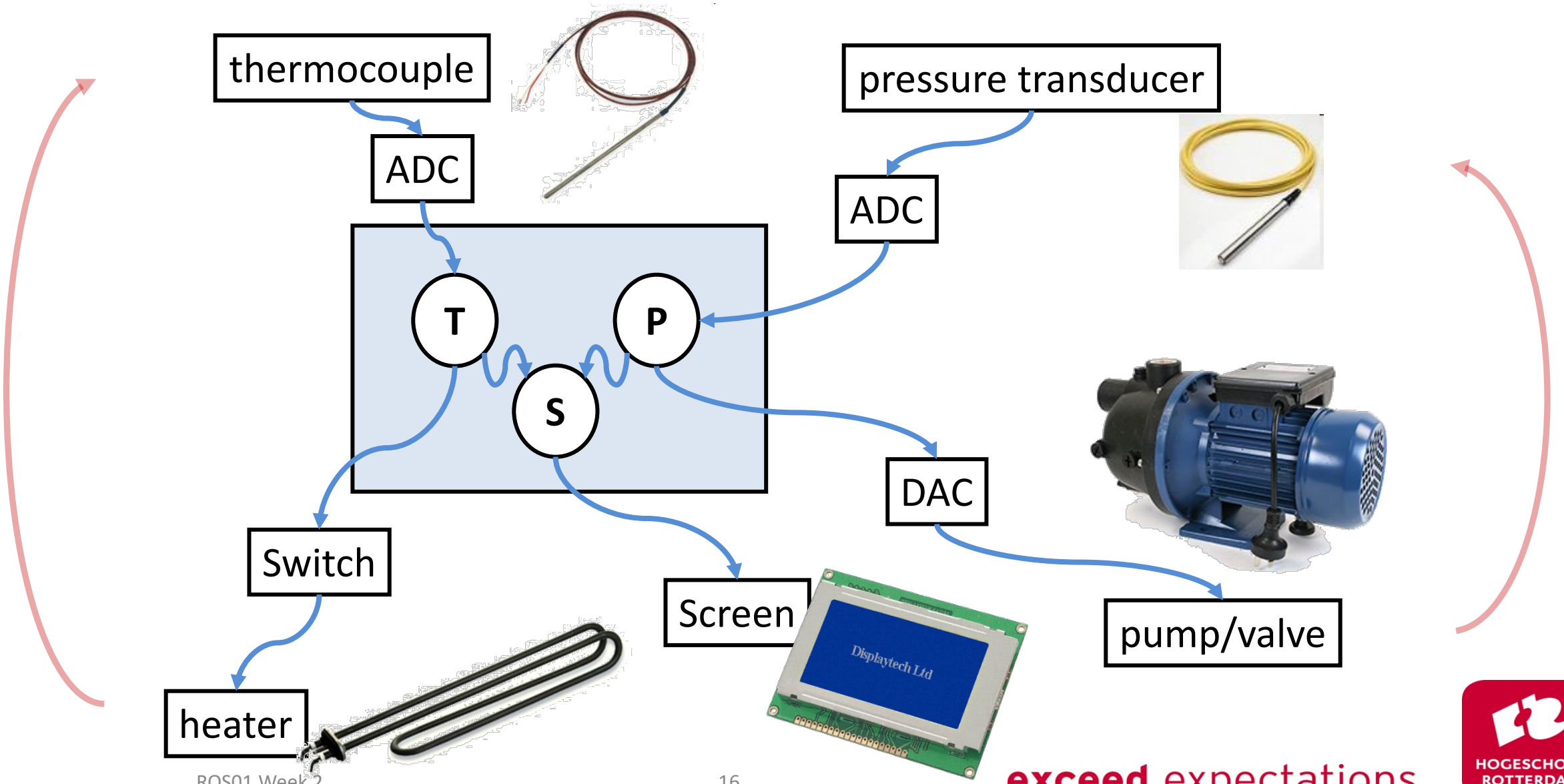
µDMA

- (Bulk) data-transfers without CPU (32 channels)
  - Memory <-> Memory
  - Memory <-> Peripheral
- Several transfer modes
  - Basic
  - Ping-pong (two alternating buffers to support continuous data flow)
  - Scatter-gather
- Configuration
  - Channel control structures

Table 4-2. Individual Control Structure

Offset	Description
0x000	Source end pointer
0x004	Destination end pointer
0x008	Control word
0x00C	Reserved

Voorbeeld embedded system

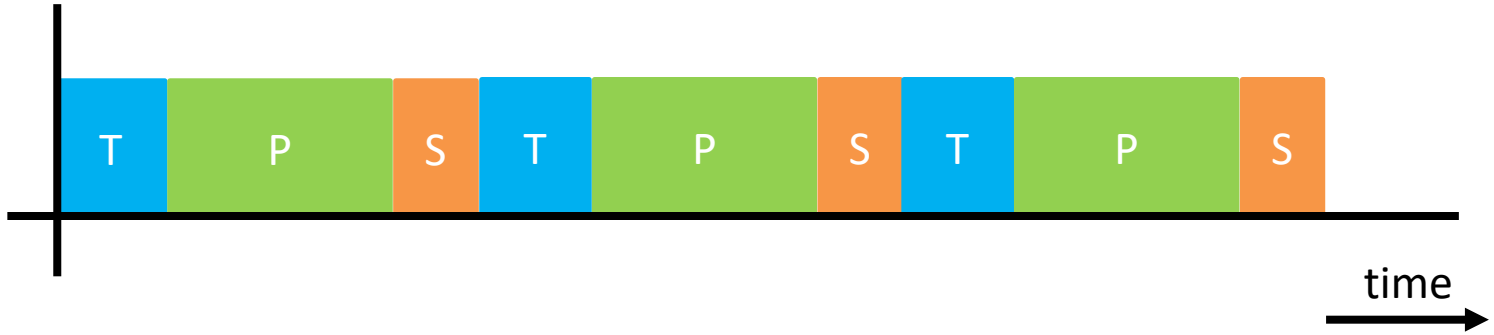


Software Architecture

What is the biggest flaw with this architecture?

- NO control over time
  - Not deterministic
    - e.g. sample time of T and P depends on execution times
  - No response time promises
  - Wasting CPU cycles

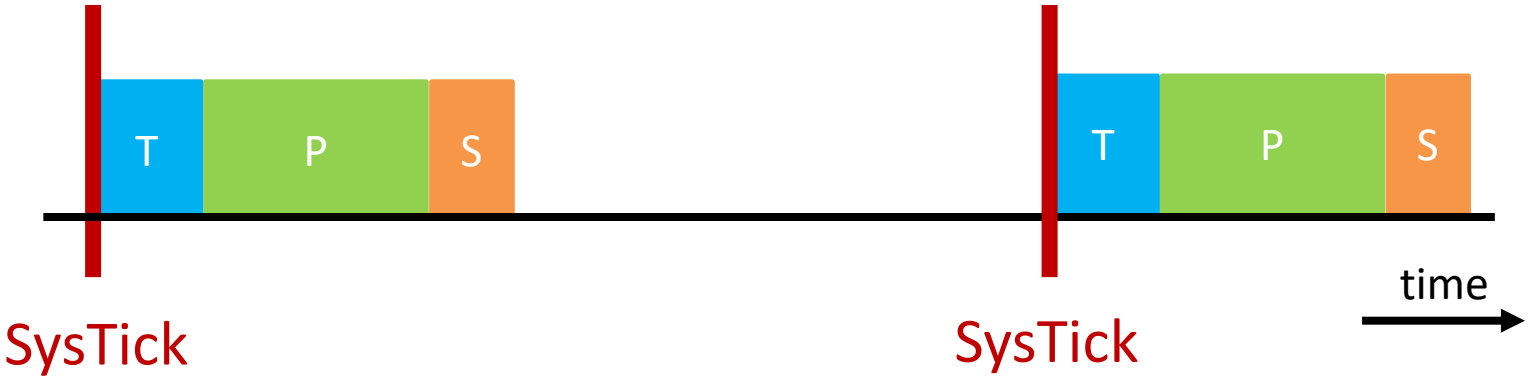
T = temperature control
P = pressure control
S = screen update



Superloop Construct

- Simple, deterministic
- Fixed time scheduling using SysTick
- Sleep until next SysTick (save energy)

- Is it necessary to run all tasks every tick?



Opdrachten / Verslag

- Opdrachten worden niet afgetekend
  - Komen wel in het verslag
- Je mag om feedback vragen

- Verslag
  - Bevat code opdrachten
  - Uitleg interessante delen
    - Bijv. Het adres-masker bij week 1