

# IAR Systems Day - ARM Tech Con 2010

**10:30am - 11:20 am**

## **ARM Microcontroller Debug and Trace Infrastructure**

Rob Cosaro, NXP and Anders Lundgren, IAR Systems

**11:30am - 12:20 pm**

## **Taking Trace to the Next Level**

Shane Titus, SEGGER and Lotta Frimanson IAR Systems

**2:30pm - 3:20 pm**

## **Important Technology Factors for Energy Efficient Microcontroller Design**

Raman Sharma, Energy Micro and Anders Lundgren, IAR Systems

**3:30pm - 4:20pm**

## **Debugging in the Power Domain**

Lotta Frimanson, IAR Systems and Anders Lundgren, IAR Systems

**4:30pm - 5:20pm**

## **The Inefficiency of C++, Fact or Fiction?**

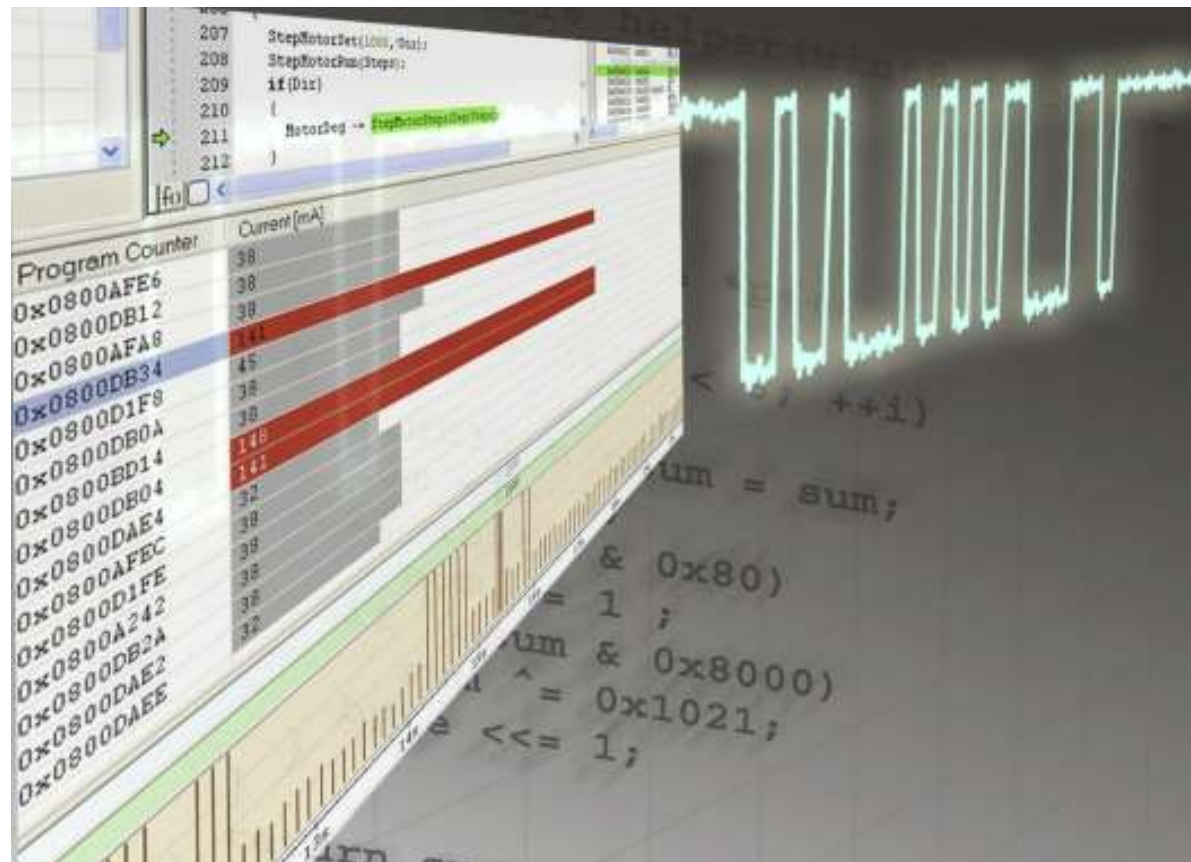
Anders Lundgren, IAR Systems



# Debugging in the power domain

IAR Systems

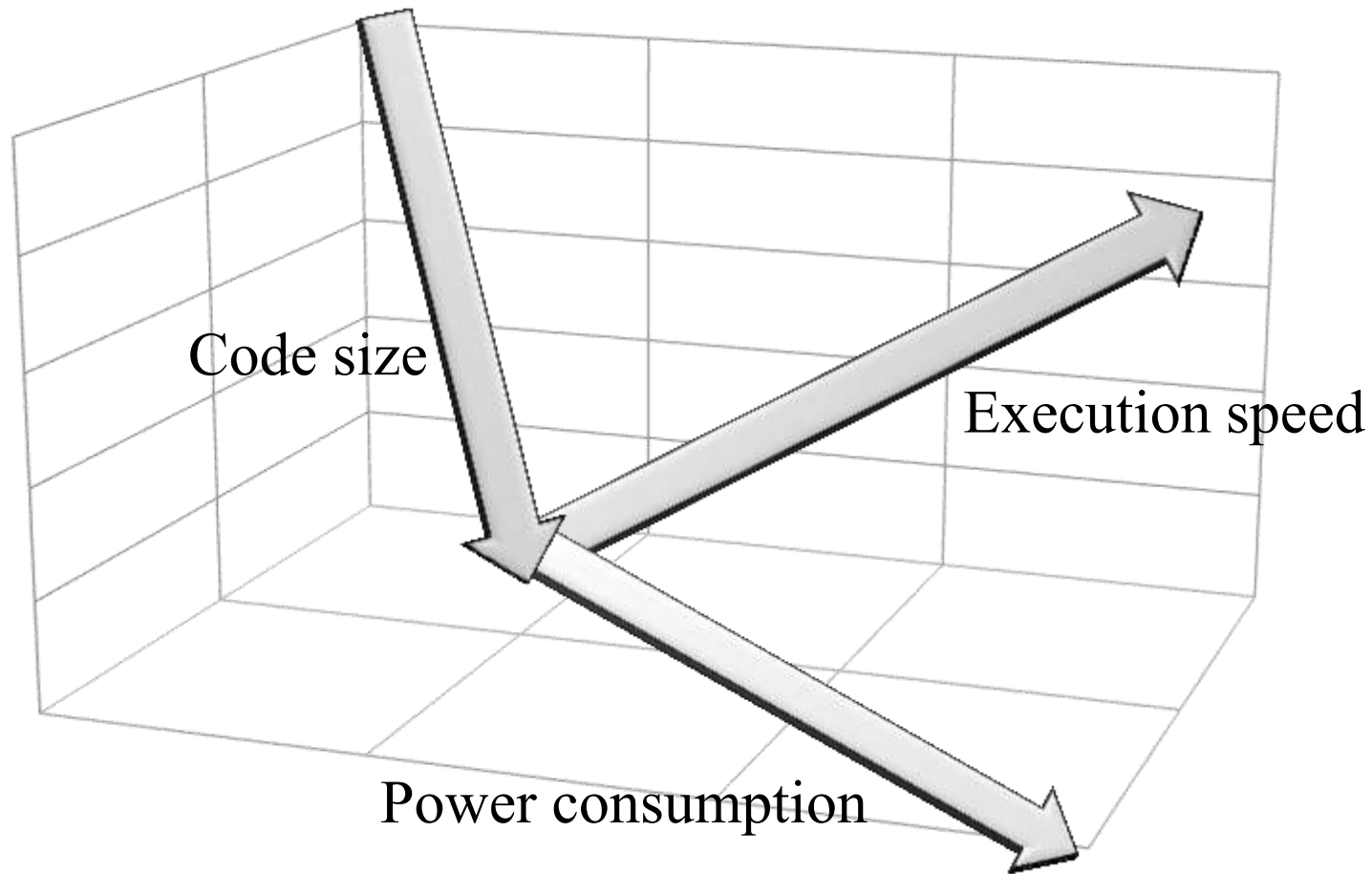
Anders Lundgren  
Lotta Frimanson




# Agenda

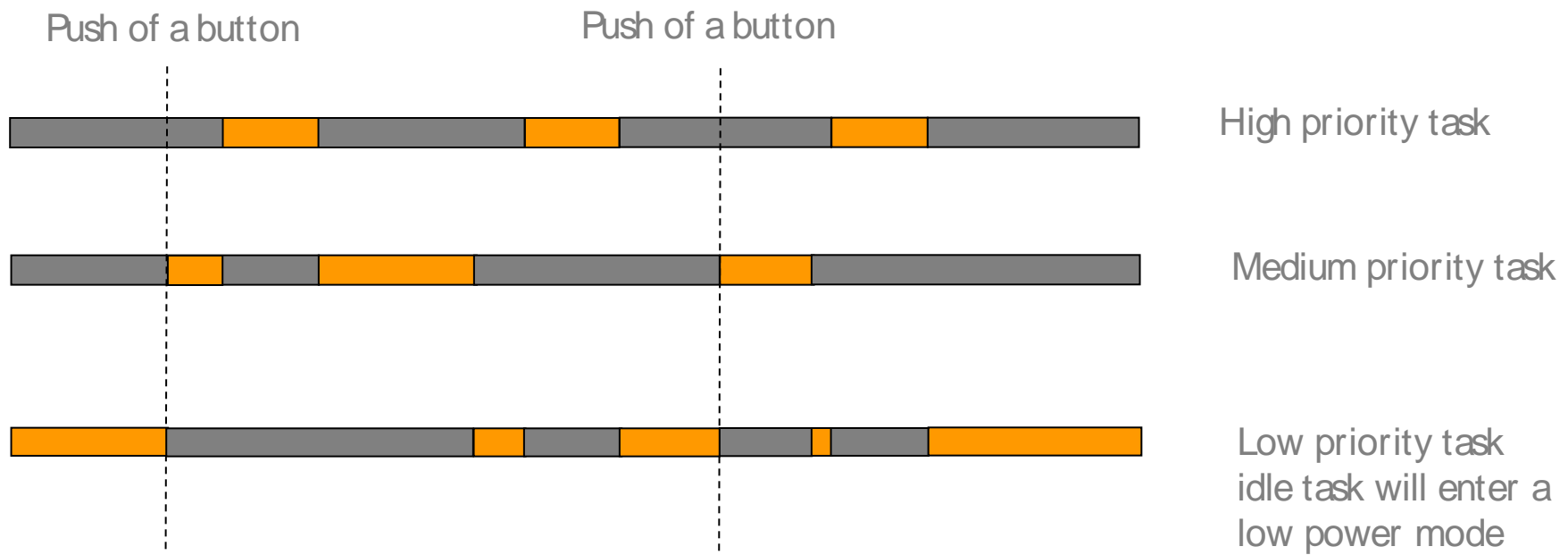
- Introduction to power debugging
- The technique behind
- Optimizing code for power
- Demonstration of power debugging features
-

# The third dimension



# Compiler power optimization

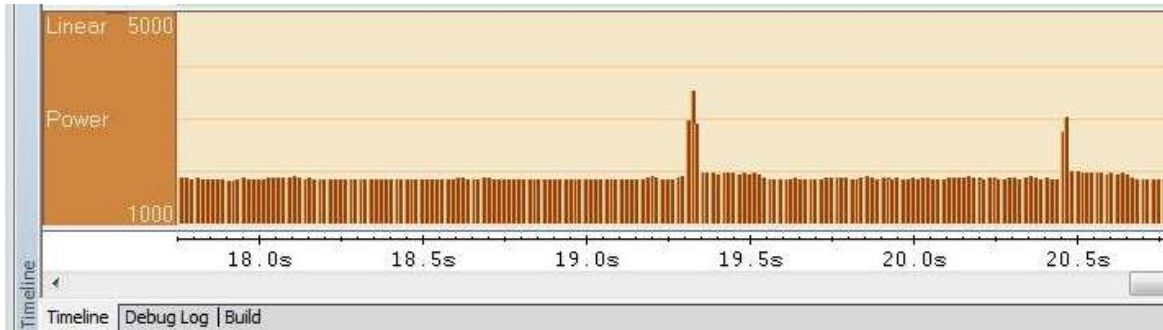
- Maximizing idle time  reducing power consumption
- The faster a task is executed, the more time can be spent in a low power mode.
- Optimizing for power is very similar to optimizing for speed



# What is power debugging?

- The technology for power debugging is based on the ability to sample the power consumption and correlate each sample with the program's instruction sequence and hence with the source code.
- Source code debugger that visualizes power consumption data both statically and dynamically in different views.
- Provides a view of the power profile of an application.

# Adding power measurement to the debugger





# Who needs it?

- Applications where long battery life-time is important but also all systems that want to minimize the power consumption.
- Power consumption has traditionally been a design goal that only hardware developers have been able to influence, using power tools like multimeter and oscilloscope.
- But in an active system, the power consumption is also dependent on how the hardware is used and controlled by software.
- Provides a tool for software developers.

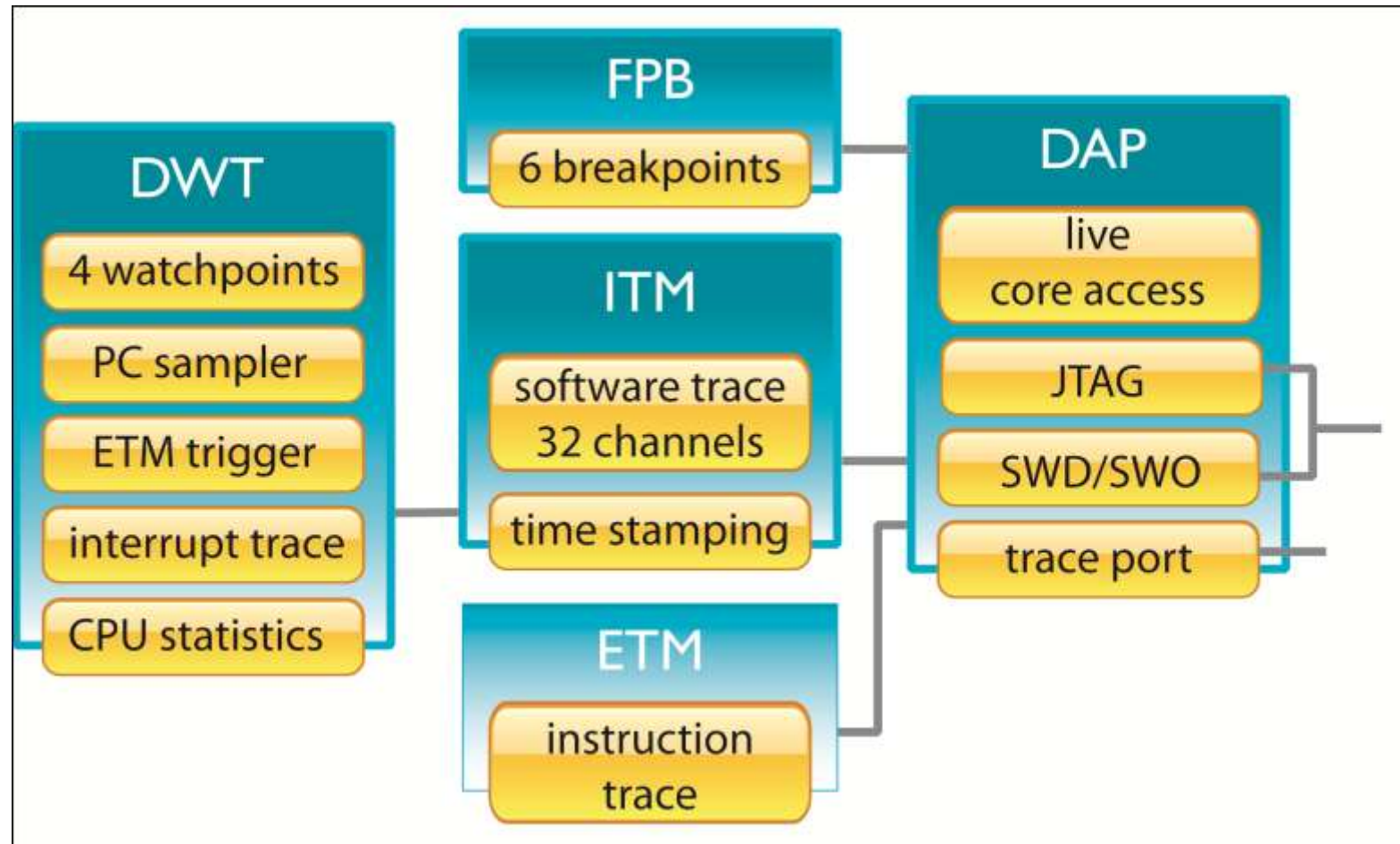




# Agenda

- Introduction to power debugging
- The technique behind
- Optimizing code for power
- Demonstration of power debugging features

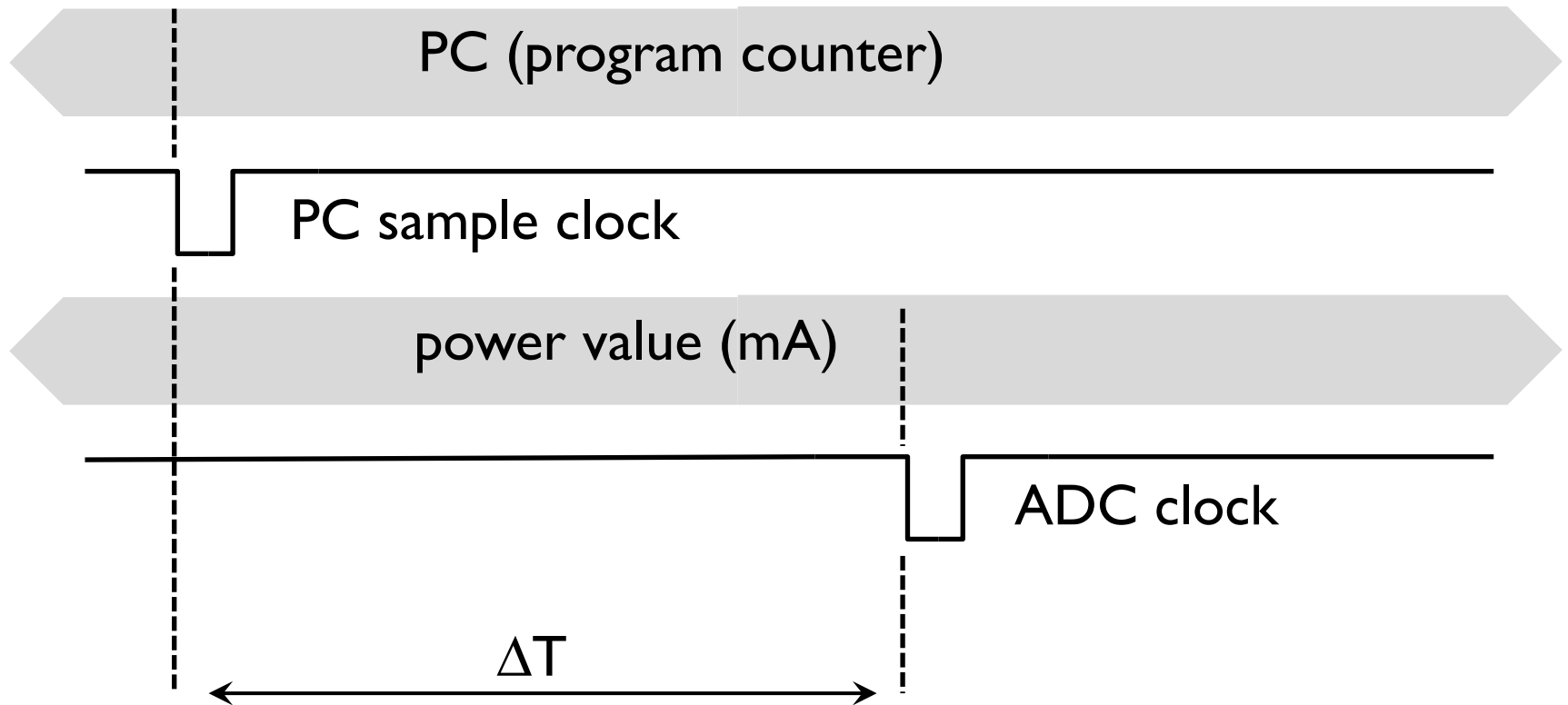
# Case study: Cortex-M3/M4



# How does it work?

- The PC sampling facility in the DWT module available in the ARM Cortex-M3/M4 cores is used.
- It samples the PC (Program Counter) periodically around 5000 times per second and triggers an ITM packet for each sample taken.
- The ITM is the formatter for events originating from the DWT. It packetizes the events and timestamps them.
- The debug probe, samples the power consumption of the device using an AD converter.
- By time stamping the sampled power values and the PC samples it is possible for the debugger to present power data on the same time axis as graphs like interrupt log and variable plots, and to correlate power data to source code.

# How does it work?



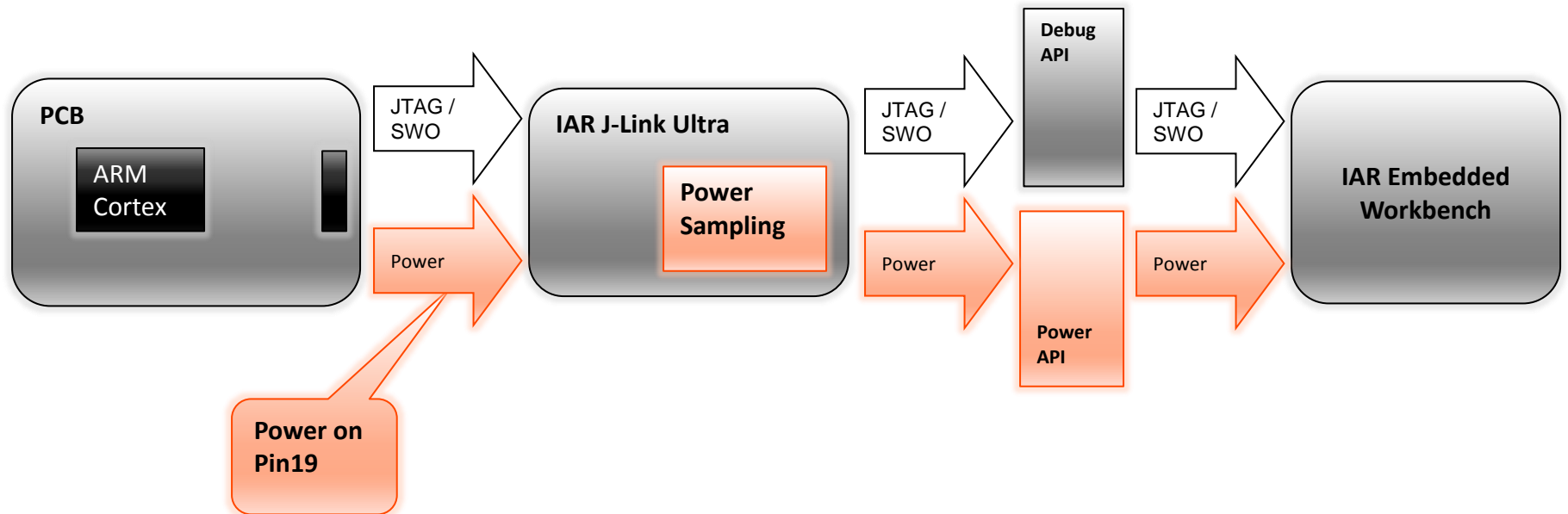
# Board versus chip level power measurement

- Board level
  - measures total system power
  - non-intrusive using JTAG pin-19
- Chip level
  - measures power at MCU VDD pins
  - intrusive

# Chip level power measurement

	Pros	Cons
Board level	<ul style="list-style-type: none"><li>○ easy to use, plug n' play with IAR boards using J-Link Ultra</li><li>○ includes the power of all components</li></ul>	<ul style="list-style-type: none"><li>○ power reading distorted by non-relevant components</li><li>○ slow response due to voltage regulators and big capacitors</li></ul>
Chip level	<ul style="list-style-type: none"><li>○ “fast” response</li><li>○ can easily see how code changes affects MCUVDD power</li></ul>	<ul style="list-style-type: none"><li>○ needs multiple channels to get system effects (MCU, SDRAM, ...)</li><li>○ difficult to gain access to chip VDD feed, sometimes jumpers, otherwise cut PCB traces or de-solder pins.</li></ul>

# What we did





# J-Link Ultra



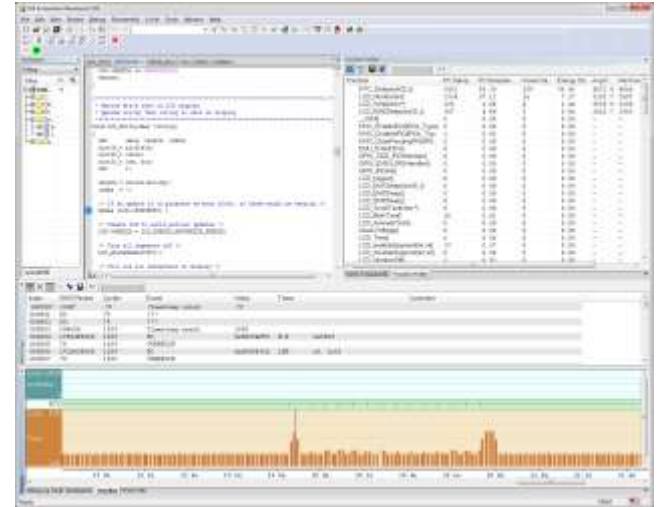
- New J-Link debug probe from IAR systems
- USB high speed (480 Mbit/s)
- ARM7/ARM9/ARM11/Cortex-M0/M1/M3/M4/A5/A8/A9 core supported, JTAG and SWD.
- Max. JTAG speed 25 MHz
- Max SWO speed 48MHz
  - Serial and Manchester decoding
- Optional 5V power on JTAG pin 19
- Power sampling on pin-19 at 1mA resolution
- Power sampling at 10 kHz (firmware upgrade)

# Power sampling performance

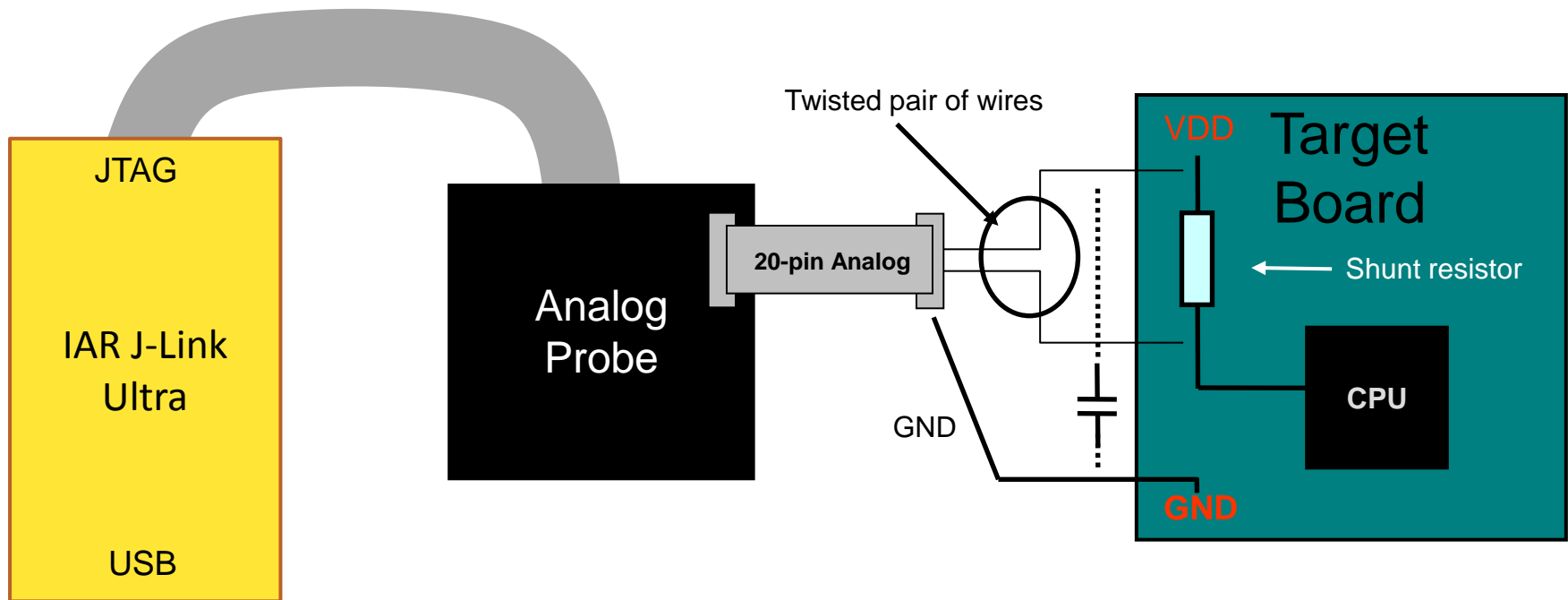
- J-Link
  - resolution: 7mA
  - accuracy: ~30mA
- J-Link Ultra
  - resolution: 1mA
  - accuracy: a few mA
- The sampling frequency is today around 100Hz
- EWARM 6.10 will sample at 10kHz
  - typical board level frequency response is ~1000Hz.

# Power debugging features

- Visualization with Power graph in Timeline window
  - Provides a visual view of the applications power profile
- Statistical power profiling; energy percentage, average, min and max values are provide in the Function Profiler window
  - Identifies the functions that consume most power in the application.
- Correlation to program counter and by that with the running application.
  - Double click in the Power graph to find the corresponding source code.
- Power log window provides textual log of all power samples together with timestamp and program counter.



# The next step – higher resolution



Sample frequency: 50 kHz

Resolution: a few  $\mu\text{A}$

Accuracy: a few  $\mu\text{A}$

# Agenda

- Introduction to power debugging
- The technique behind
- Optimizing code for power
- Demonstration of power debugging features

# Optimizing code for power

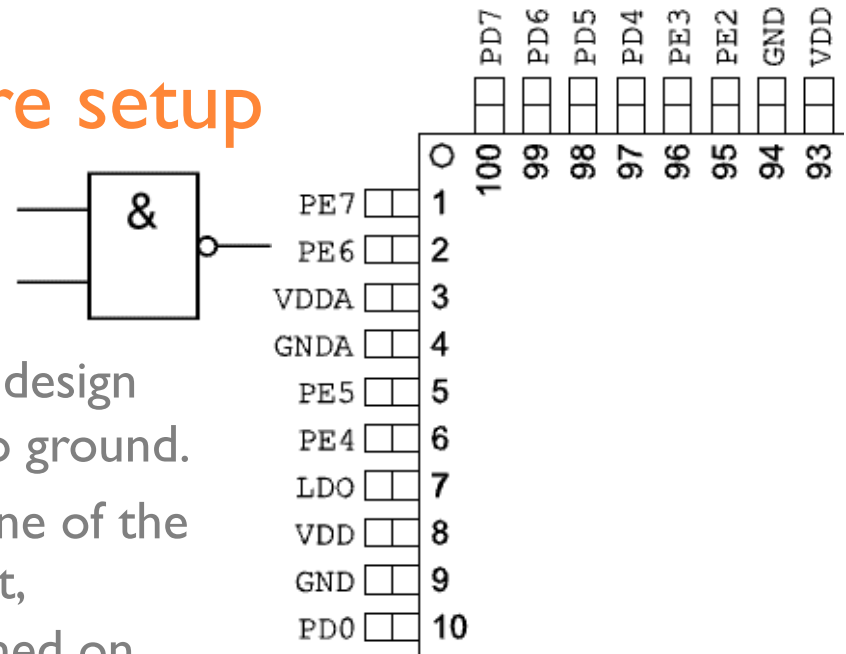
- Identifying unnecessary energy consumption in a system is difficult.
- Typically it is not explicit flaws in the source code that is exposed, but rather opportunities to tune how the hardware is utilized.
- Sometimes, however, it may involve what might be described as pure bugs.

# DMA vs polled I/O

- DMA has traditionally been used to increase transfer speed.
- In the MCU world there are a large number of DMA techniques to increase flexibility, speed, and to lower power consumption.
- In some architectures the CPU can even be put into sleep mode during the DMA transfer.
- Power debugging allows the developer to experiment and see directly in the debugger what effects these DMA techniques will have compared to a traditional CPU driven polled approach.

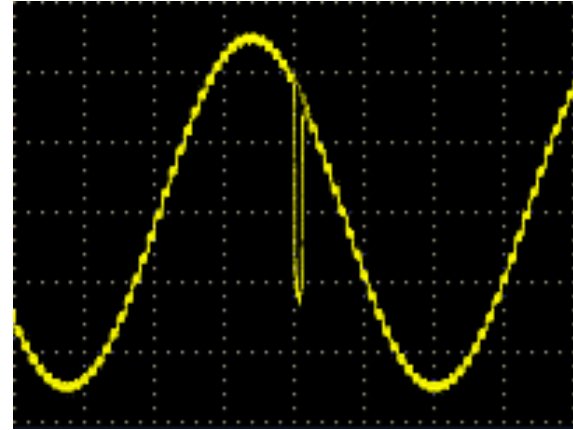


# Finding conflicting hardware setup



- To avoid floating inputs it is a common design practice to tie unused MCU I/O pins to ground.
- If the software by mistake configures one of the grounded I/O pins as a logical '1' output, a current as high as 25mA may be drained on that pin.
- This high unexpected current is easily observed by reading the current value from the power graph; it is also possible to find the corresponding erratic initialization code by looking at the power graph at application startup.
- A similar situation will arise if an I/O pin is designed to be an input and is driven by an external circuit, but the software incorrectly configures the input pin as output.

# Analog interference



- Mixing analog and digital circuits on the same board has its own challenges and board layout and routing becomes important to keep the analog noise at a low level.
- Software design can also affect the quality of the analog measurements. Performing I/O activity, for example pulsing a stepper motor, at the same time as sampling analog signals can cause spikes giving wrong AD converter samples.

# Waiting for device status

Common mistakes that could cause unnecessary power to be consumed:

- Poll loop to wait for a status change of for example a peripheral device.

```
while ((BASE_PMC->PMC_SR & MC_MCKRDY) != PMC_MCKRDY);
```

Use low power mode and interrupt instead!

- Software delay as a for or while loop

```
i = 10000;  
do i--;  
while (i != 0);
```

Use HW timer with low power mode!

# Low power mode diagnostics

- Many embedded applications spend most of their time waiting for something to happen; receiving data on a serial port, watching an I/O pin change state, or waiting for a time delay to expire.
- Many microprocessors have a number of different low power modes, in which different parts of the processor can be turned off when they are not needed.
- By placing it in a low power mode during the idle time, the battery life can be extended by orders of magnitude.
- In a task-oriented design the idle task is the perfect place to implement power management.
- A power debugging tool can be very useful when elaborating with different low-level modes.

# CPU frequency

- Power consumption in a CMOS MCU is theoretically given by the formula:

$$P = f \times U^2 \times k$$

where:  $f$  is the clock frequency

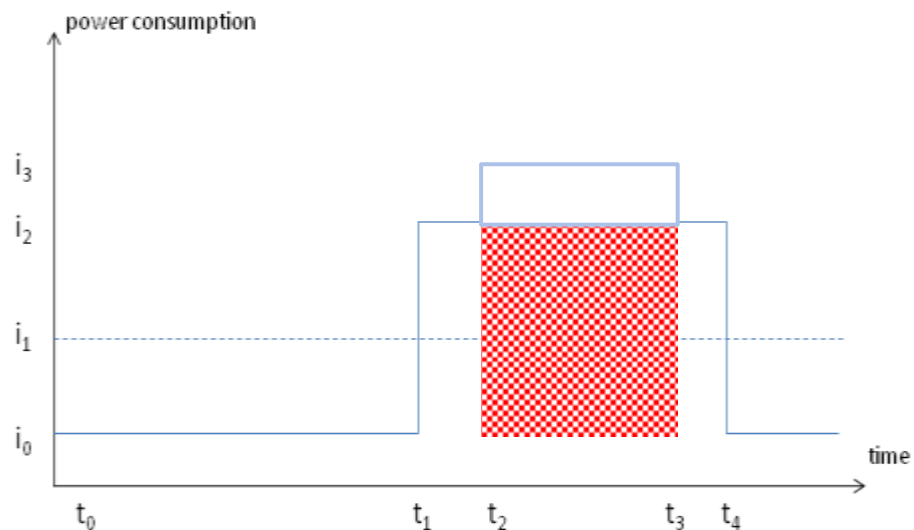
$U$  is the supply voltage

$k$  is a constant

- Reducing the clock frequency will give a proportional reduction of power consumption.
- On the other hand a high clock frequency will increase the time spent in low power mode.
- The power data in the debugger will help the developer chose the operating frequency that gives the lowest power consumption.

# Interrupt handling

- Power debugging can be used to discover an extraordinary increase in power consumption.
- An example involving interrupts can illustrate a situation when it is difficult to identify that a system consumes unnecessary energy.



Need to review whether it is worth to spend extra clock cycles to turn on and off peripherals in a situation like this.

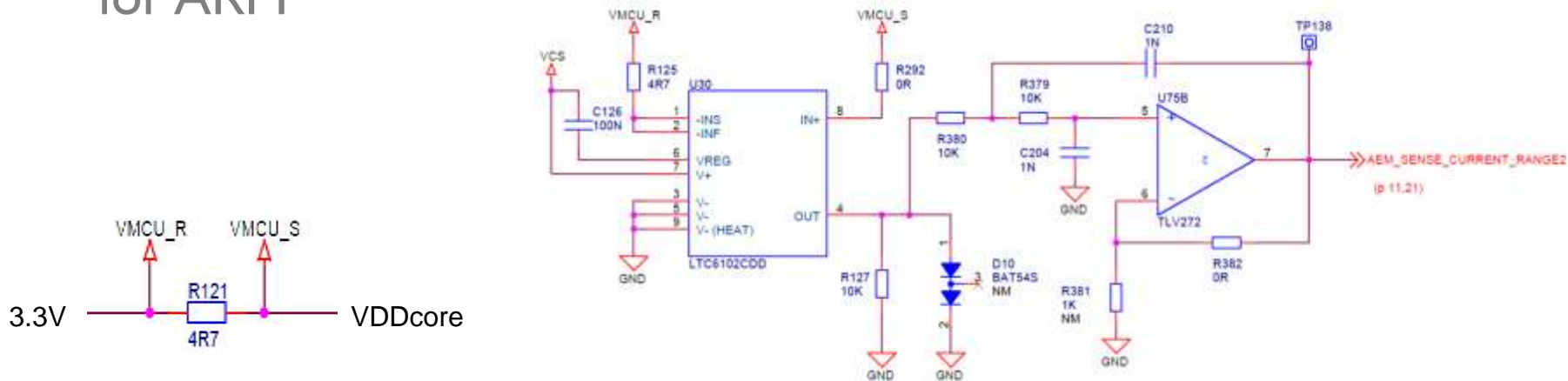
# Agenda

- Introduction to power debugging
- The technique behind
- Optimizing code for power
- Demonstration of power debugging features



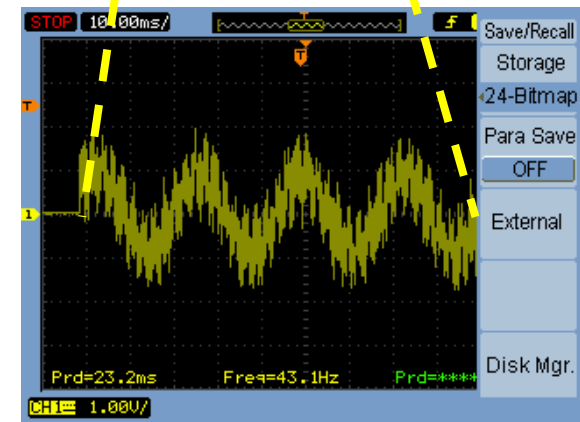
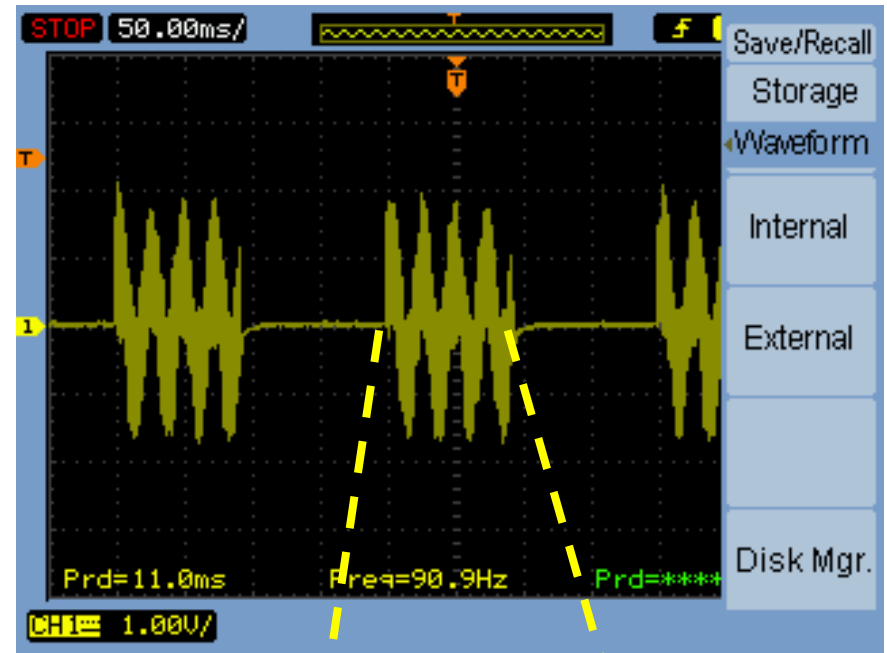
# The demonstration setup

- EFM32 Gecko Development kit from Energy Micro
- J-Link on-board from SEGGER
- IAR Embedded Workbench for ARM



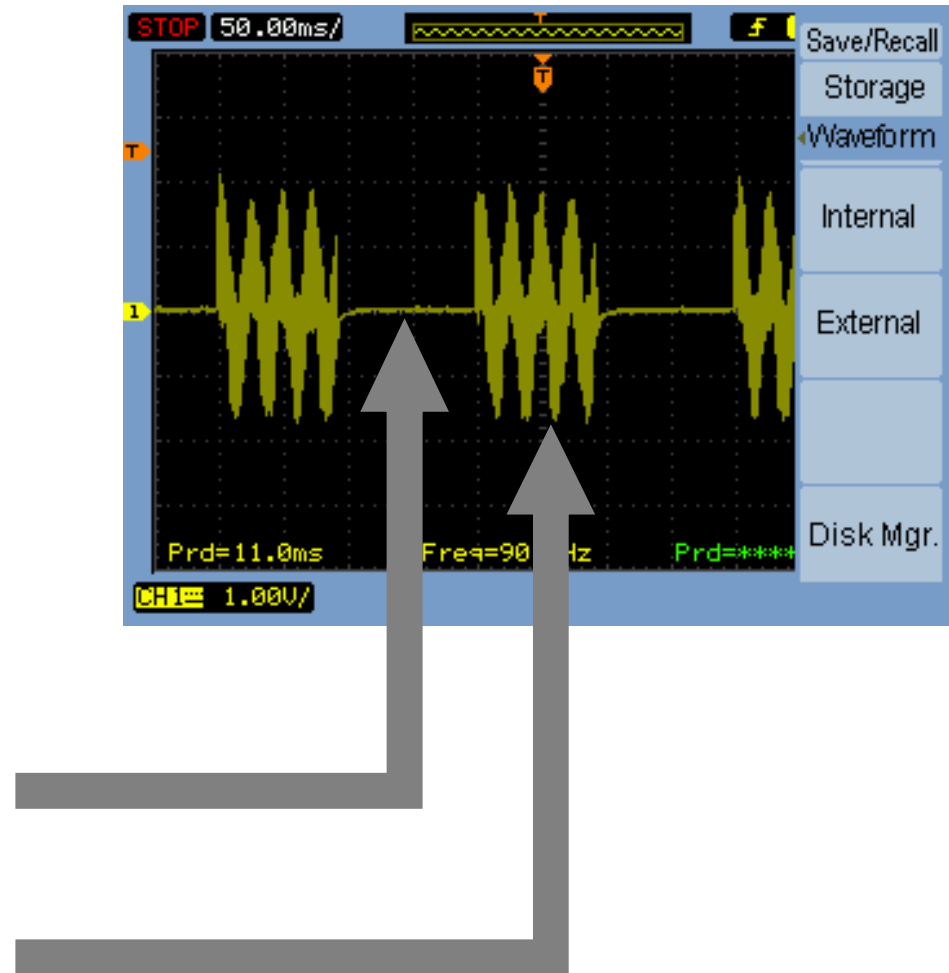
# The demo application

- Burst waveform generator for low-frequency marine sonar system
- Flexible burst waveform by using inverse FFT
- Two selectable waveforms
- 10 kHz sampling rate
- 100 ms burst length
- 0 – 500ms repetition rate



# The demo flow

- Initial attempt
- Serious bug (lots of power consumed here)
- Hot spot in RAM
- Clock frequency change
- Low power mode EM2 between bursts
- Low power mode EMI between samples



# Demonstration

Thank You!