Graceful

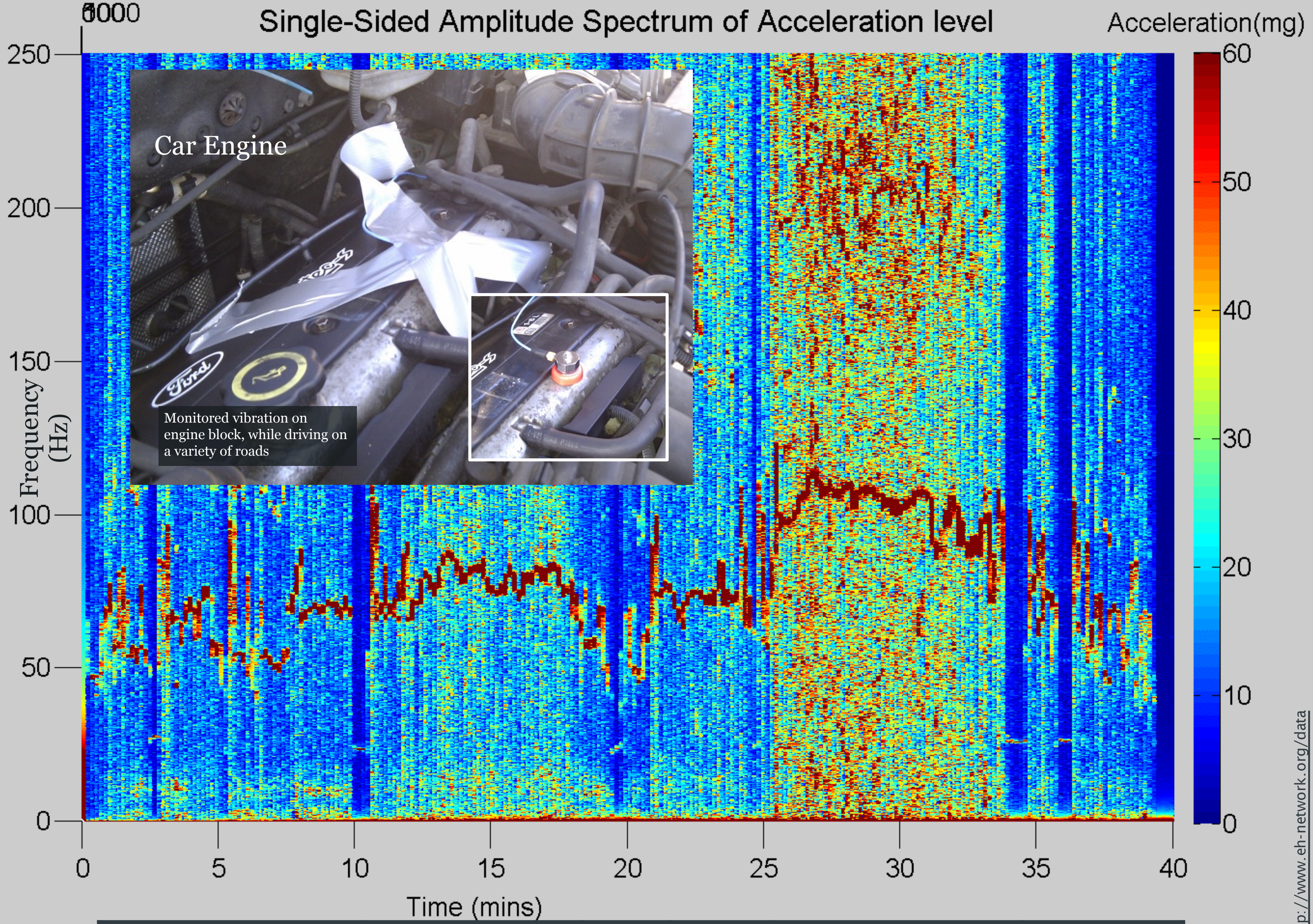
EPSRC
Pioneering research
and skills

UNIVERSITY OF
Southampton

Transient and Power-Neutral Computing:
A Paradigm Shift for Embedded Systems?

Geoff Merrett, 02 April 2016

"Hilariously Low Power Computing" Workshop
ASPLOS 2016, Atlanta GA



Car Engine

Monitored vibration on engine block, while driving on a variety of roads

mph:	30	60	20	60	70	30

http://www.eh-network.org/data

Outline

towards storage-less and power-centric systems

Energy-Harvesting Systems

Operating from harvested energy

Energy-Neutral Computing

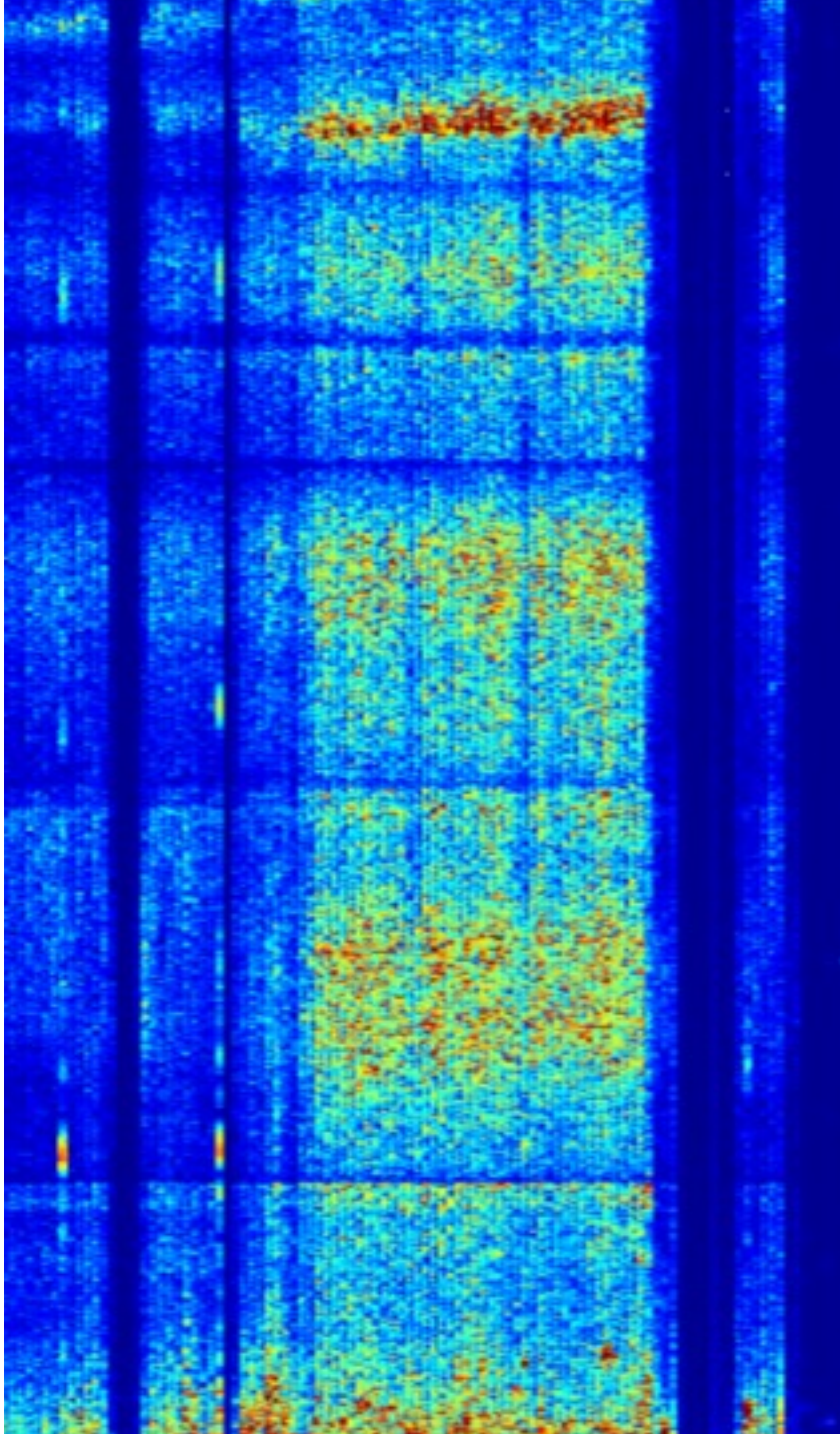
Buffering energy for a battery-like supply

Transient Computing

Computation when power is available

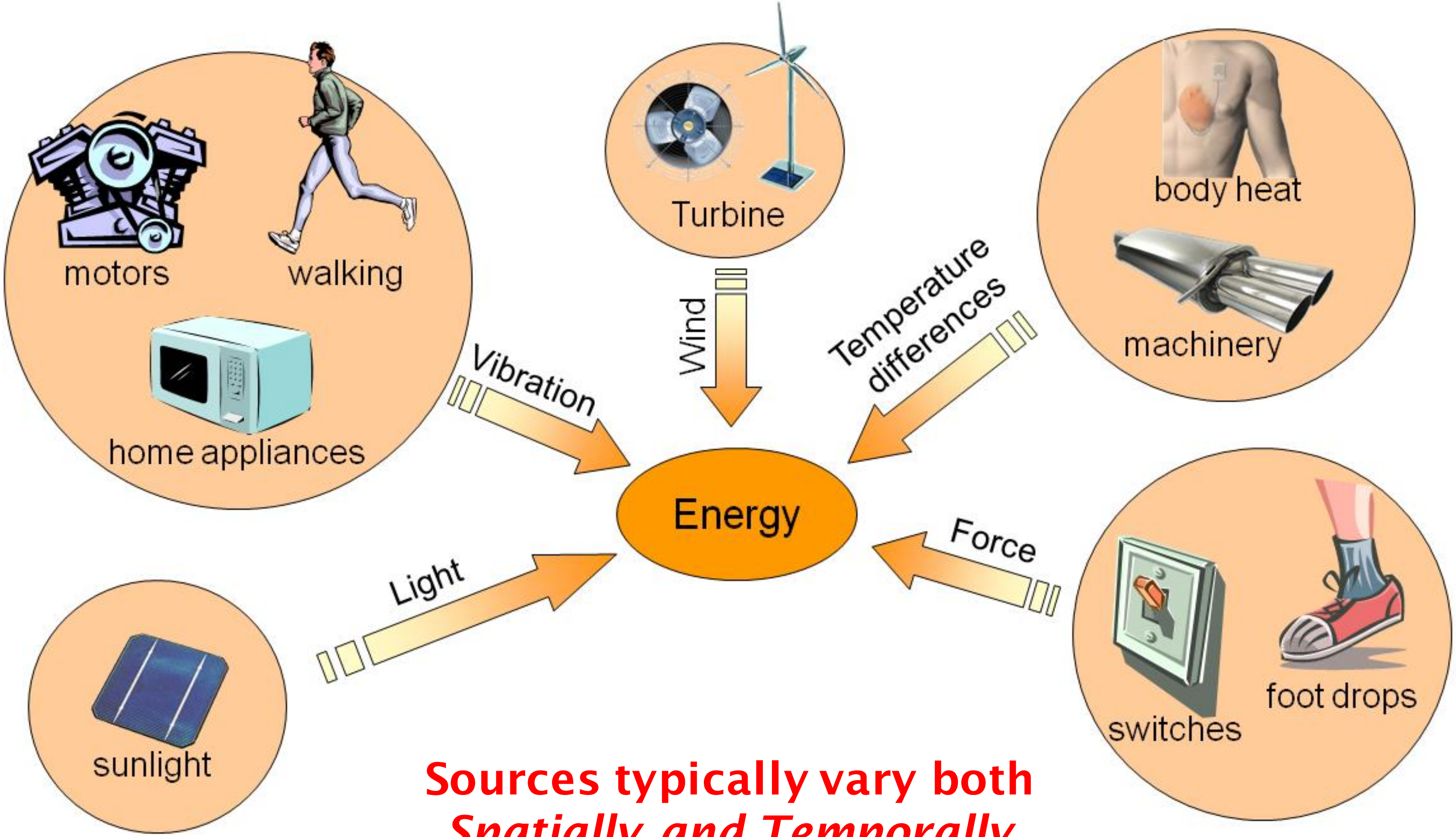
Power-Neutral Computing

Adaptive computation when power is available



UNIVERSITY OF Southampton

Energy Harvesting



Sources typically vary both
Spatially and Temporally

Energy harvesting

Irregular and unpredictable power generation

Autonomy

Irregular and unpredictable power consumption *enable computation to be sustained despite an intermittent supply*

Outline

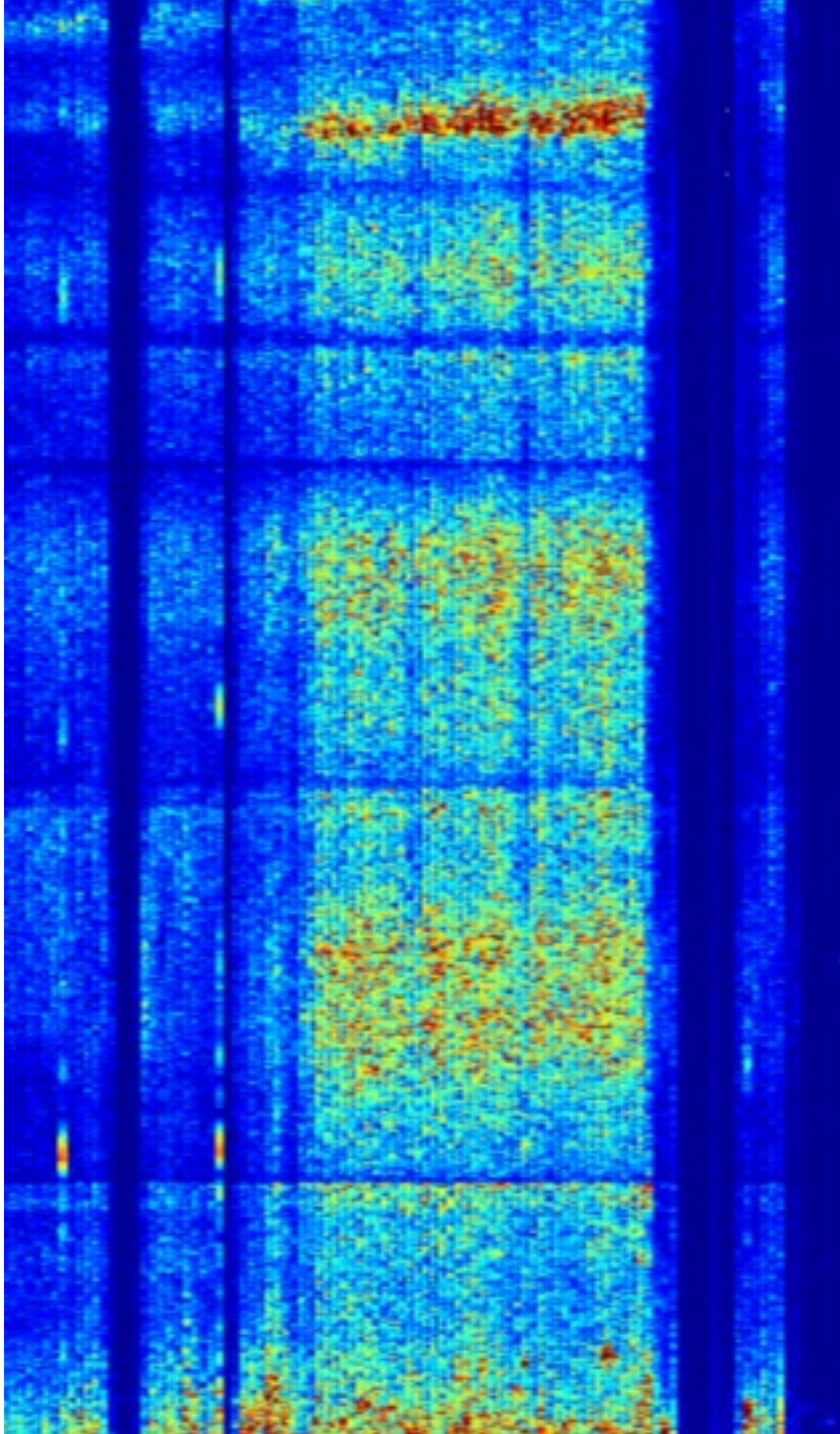
towards storage-less and power-centric systems

Energy-Harvesting Systems

Operating from harvested energy

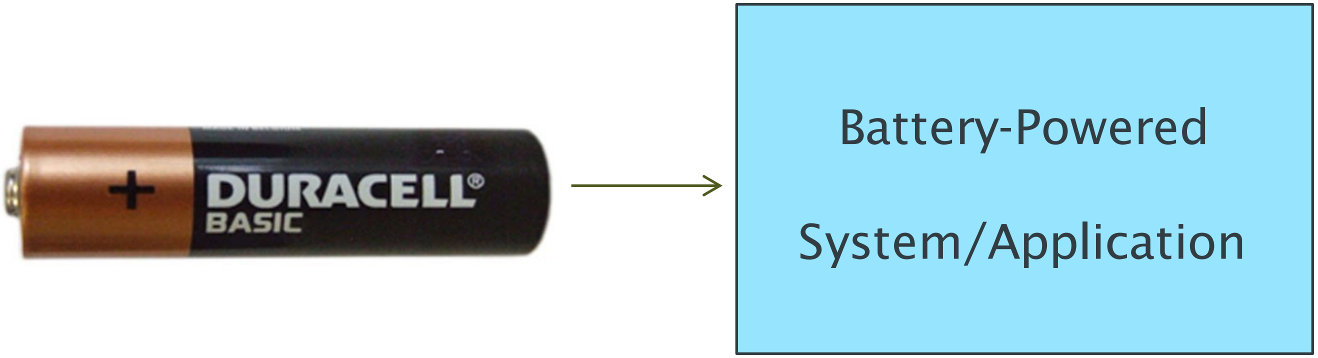
Energy-Neutral Computing

Buffering energy for a battery-like supply



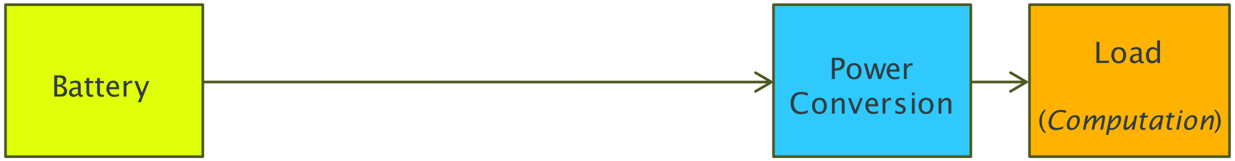
Energy-Neutral Computing

- Traditional embedded systems want to be run from a *battery-like* power supply



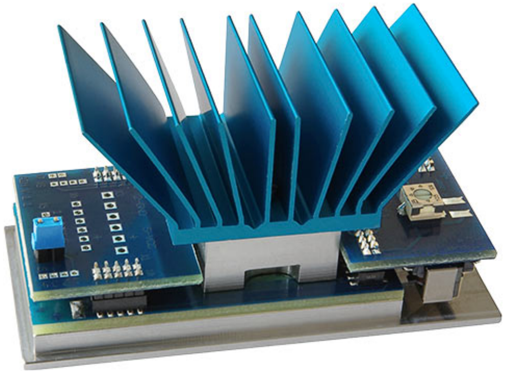
Power/current (virtually) unlimited at time *t*

Limited amount of energy over lifetime *T*



Energy-Neutral Computing

- Traditional embedded systems want to be run from a *battery-like* power supply

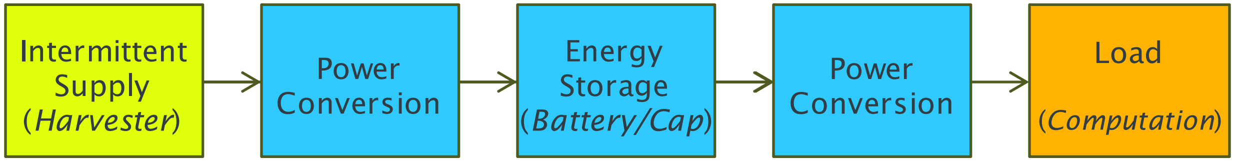


Battery-Powered

System/Application

Power/current limited (and varying/intermittent) at time *t*

(Virtually) unlimited amount of energy and lifetime *T*

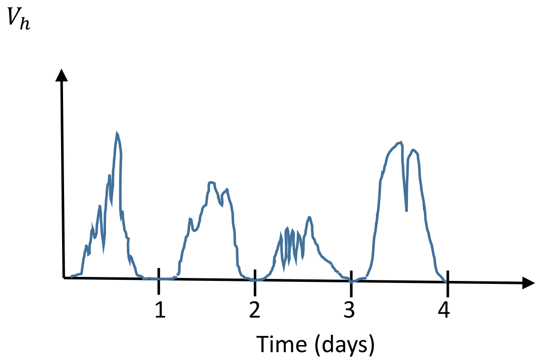


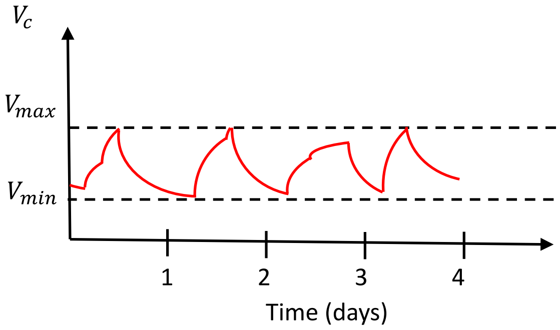
Intermittent Supply (*Harvester*) → Power Conversion → Energy Storage (*Battery/Cap*) → Power Conversion → Load (*Computation*)

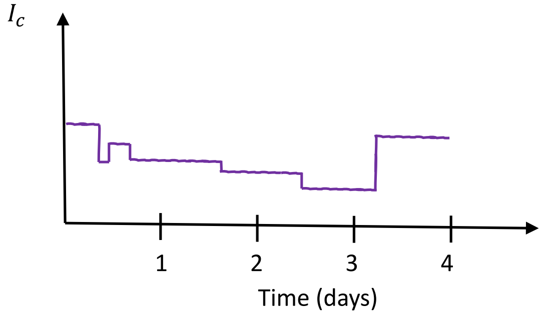
Intermittent Supply (*Mains*) → Power Conversion → Energy Storage (*Battery*) → Power Conversion → Load (*Laptop*)

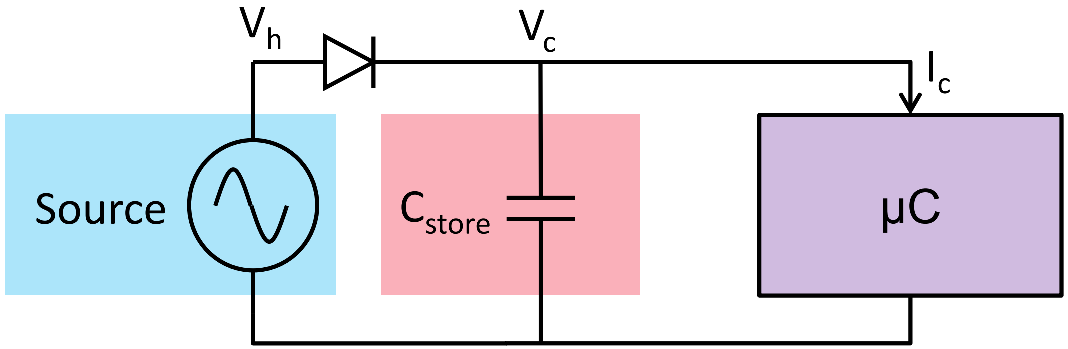
Image: http://www.mrgeek.me/wp-content/uploads/2012/12/MagSafe-2-Charging-Pin.jpg

Energy-Neutral Computing





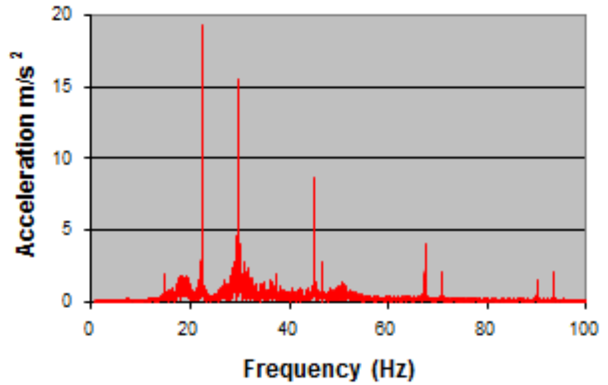


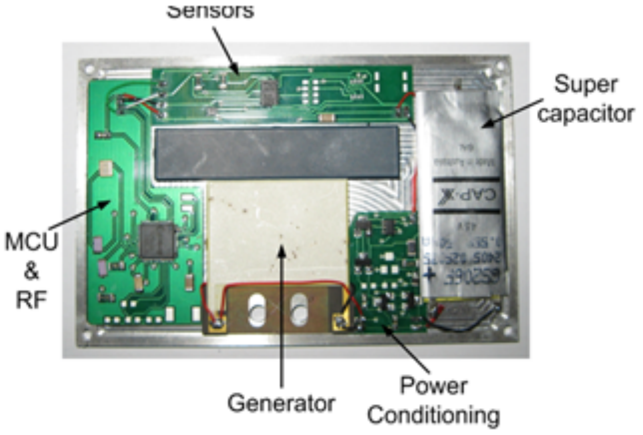


Energy Neutrality is Achieved if: $$\int_{(n-1)\cdot T}^{n\cdot T} P_h(t)\,dt = \int_{(n-1)\cdot T}^{n\cdot T} P_c(t)\,dt$$

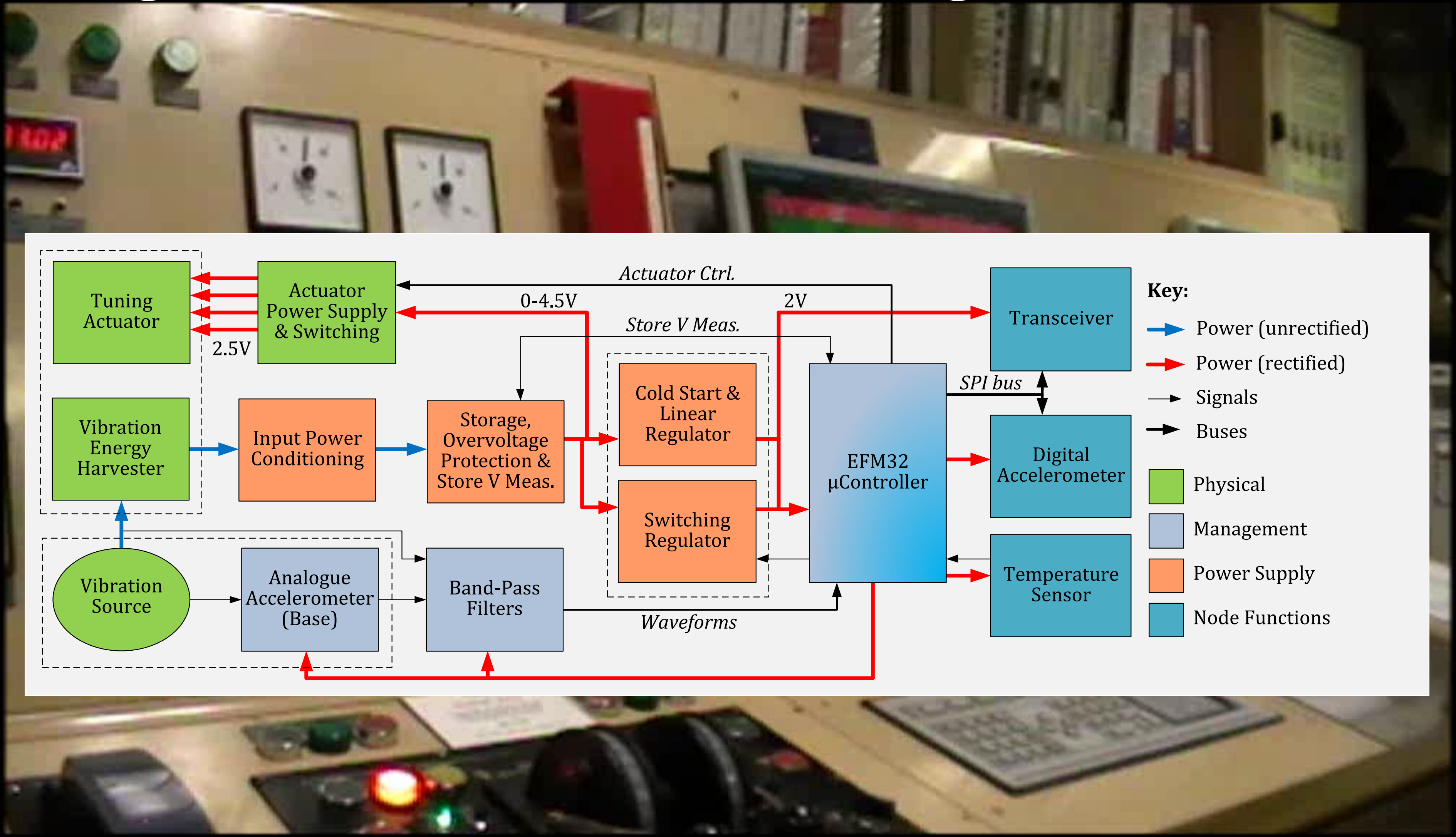
(assuming that $V_C(t) > V_{min}$ for all t)

Stamos M, Nicoleau C, Torah R, Tudor J, Harris N R, Niewiadomski A, Beeby S P Screen-Printed Piezoelectric Generator for Helicopter Health and Usage Monitoring Systems, Proc. PowerMEMS, Sendai, Japan, 2008





Engine Condition Monitoring



Tuning Actuator

Actuator Power Supply & Switching

2.5V

Vibration Energy Harvester

Input Power Conditioning

Storage, Overvoltage Protection & Store V Meas.

Actuator Ctrl.

0-4.5V

2V

Store V Meas.

Cold Start & Linear Regulator

Switching Regulator

EFM32 µController

Transceiver

SPI bus

Digital Accelerometer

Temperature Sensor

Vibration Source

Analogue Accelerometer (Base)

Band-Pass Filters

Waveforms

Key:

→ Power (unrectified)

→ Power (rectified)

→ Signals

→ Buses

🟩 Physical

▨ Management

🟧 Power Supply

🟦 Node Functions

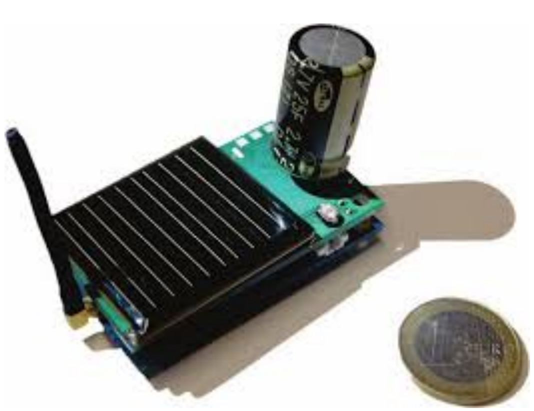
EPSRC
Pioneering research
and skills

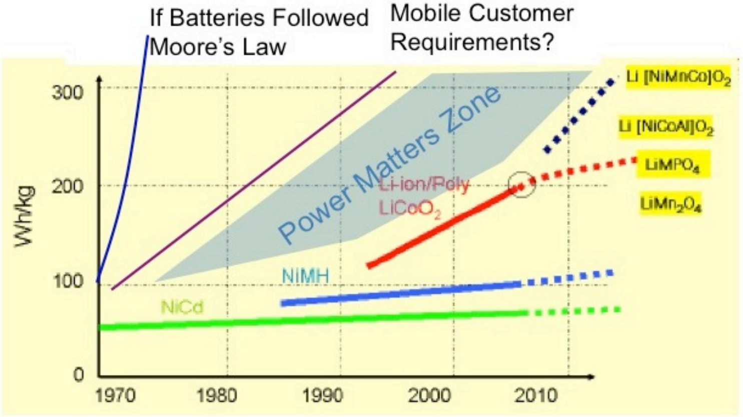
A.S. Weddell, D. Zhu, G.V. Merrett, S.P. Beeby, B.M. Al-Hashimi, (2012) A practical self-powered sensor system with a tunable vibration energy harvester. In, PowerMEMS 2012, Atlanta, US, 02 - 05 Dec 2012.

www.holistic.ecs.soton.ac.uk

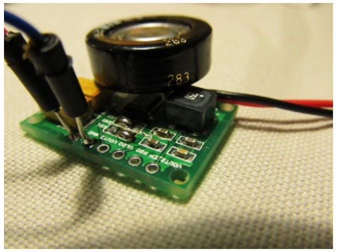
holistic
energy harvesting

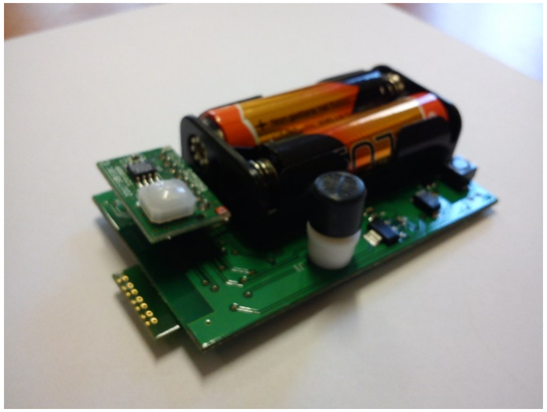
Energy-Neutral Computing





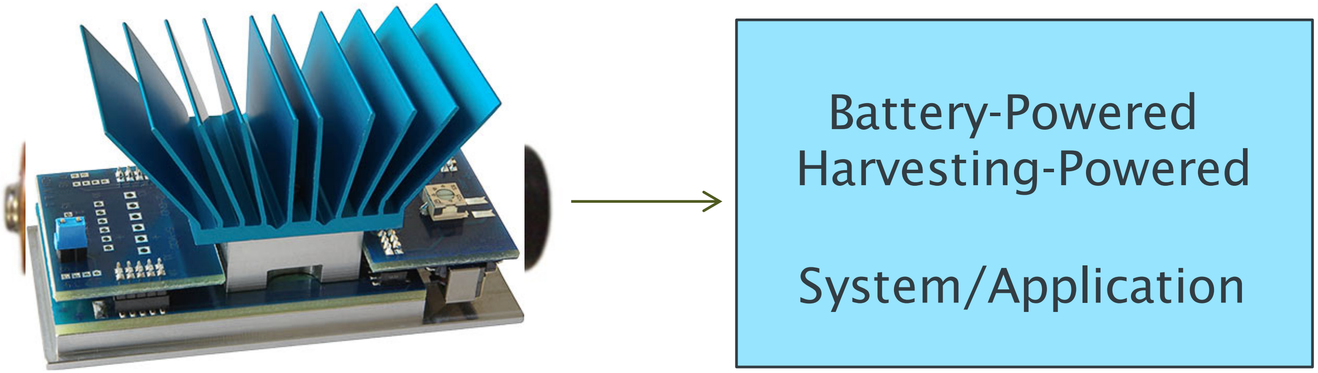
Source: Avicenne





Energy Harvesting Systems

- Don’t try to make the energy harvester look like a battery…



- …but rethink the design of energy harvesting systems.

- *How do we design energy harvesting:*
  - *computation?*
  - *communication?*
  - *electronics?*
  - *systems?*
  - *applications?*

enable computation to be sustained despite an intermittent supply…

…without the use of additional energy storage (e.g. supercapacitors or batteries)

Outline

towards storage-less and power-centric systems

Energy-Harvesting Systems

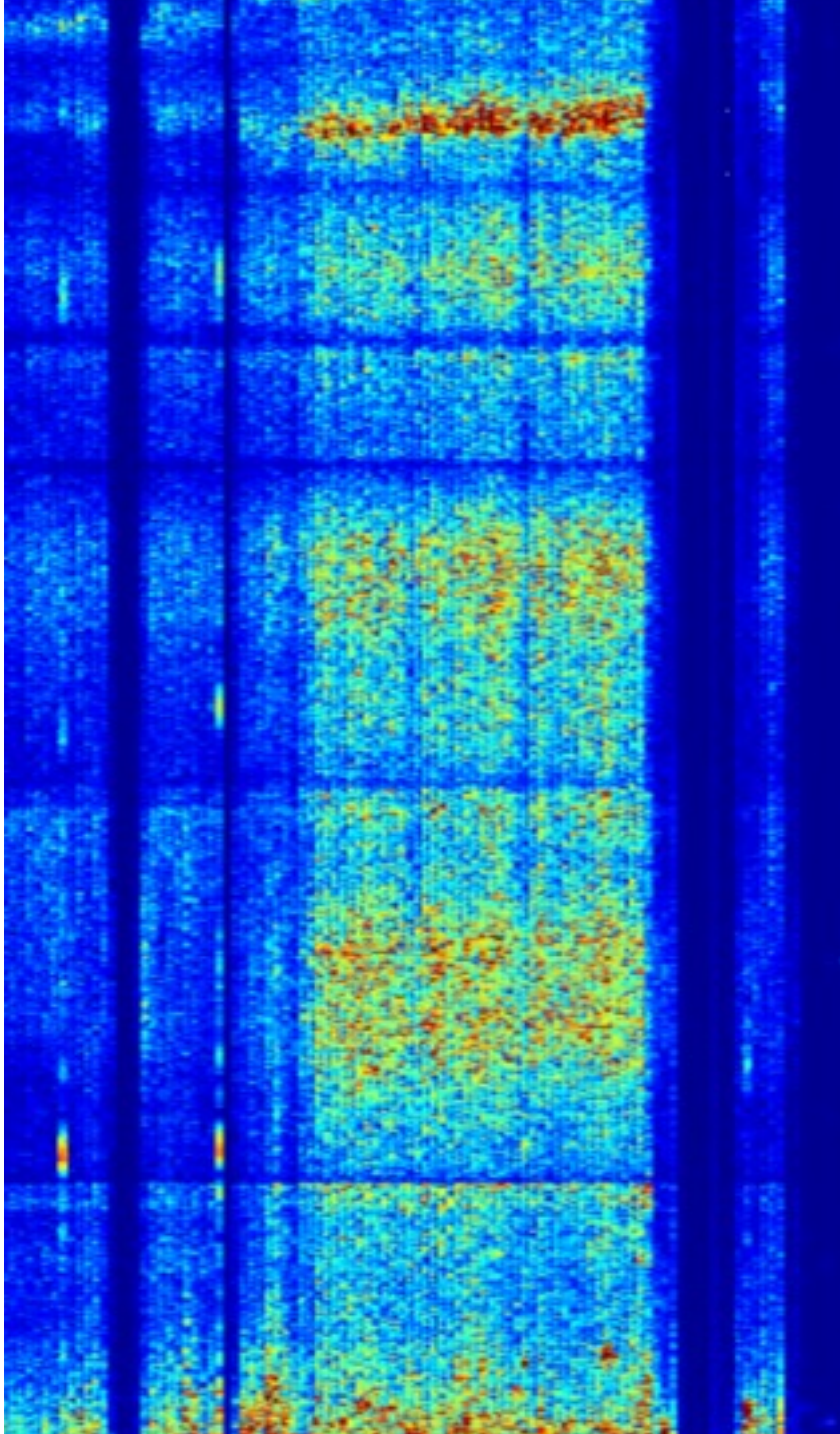
Operating from harvested energy

Energy-Neutral Computing

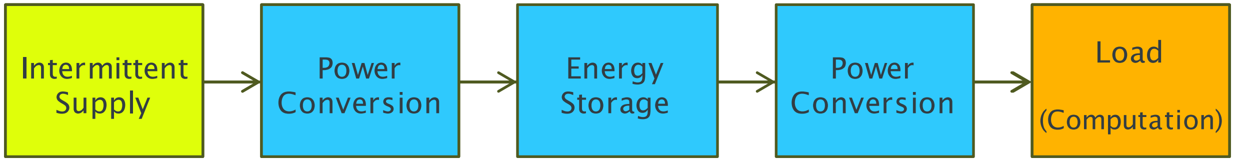
Buffering energy for a battery-like supply

Transient Computing

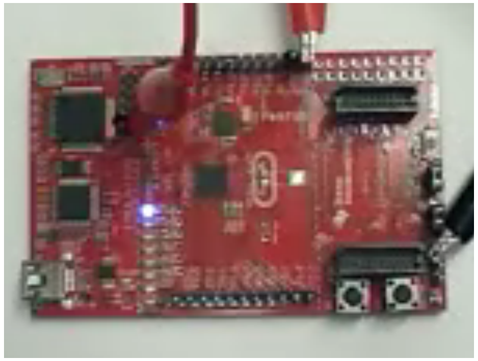
Computation when power is available

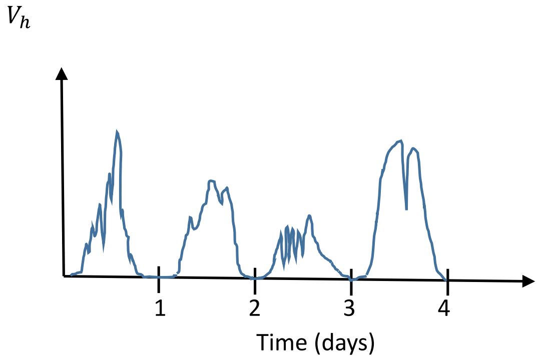


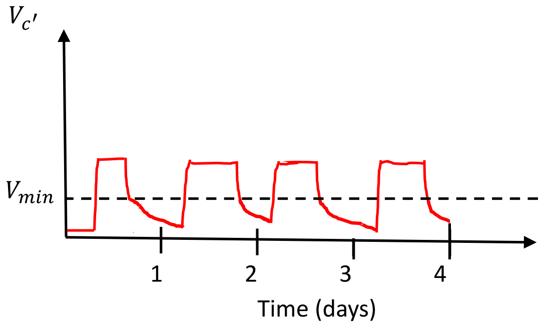
Transient Computing

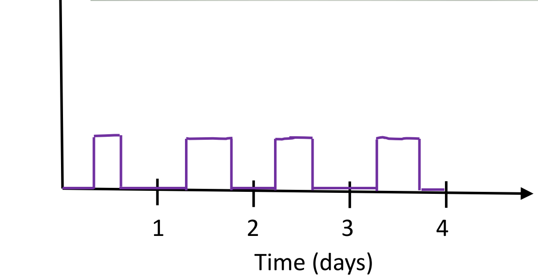


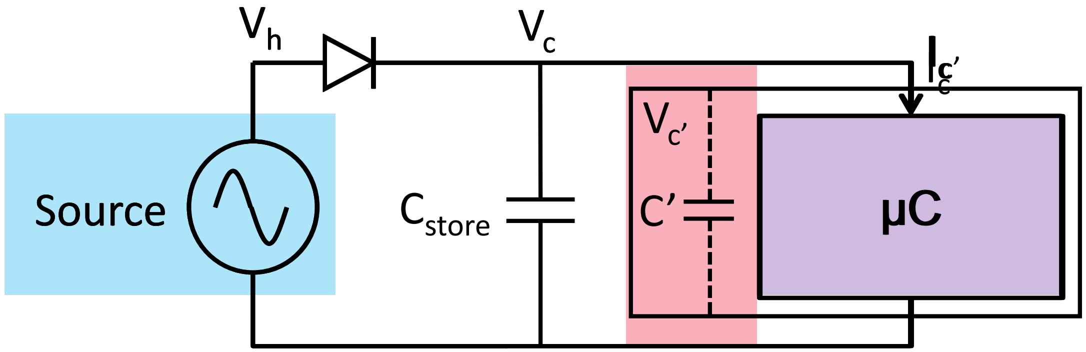
Intermittent Supply → Power Conversion → Energy Storage → Power Conversion → Load (Computation)











- How can computation accommodate this intermittent supply?

Mementos

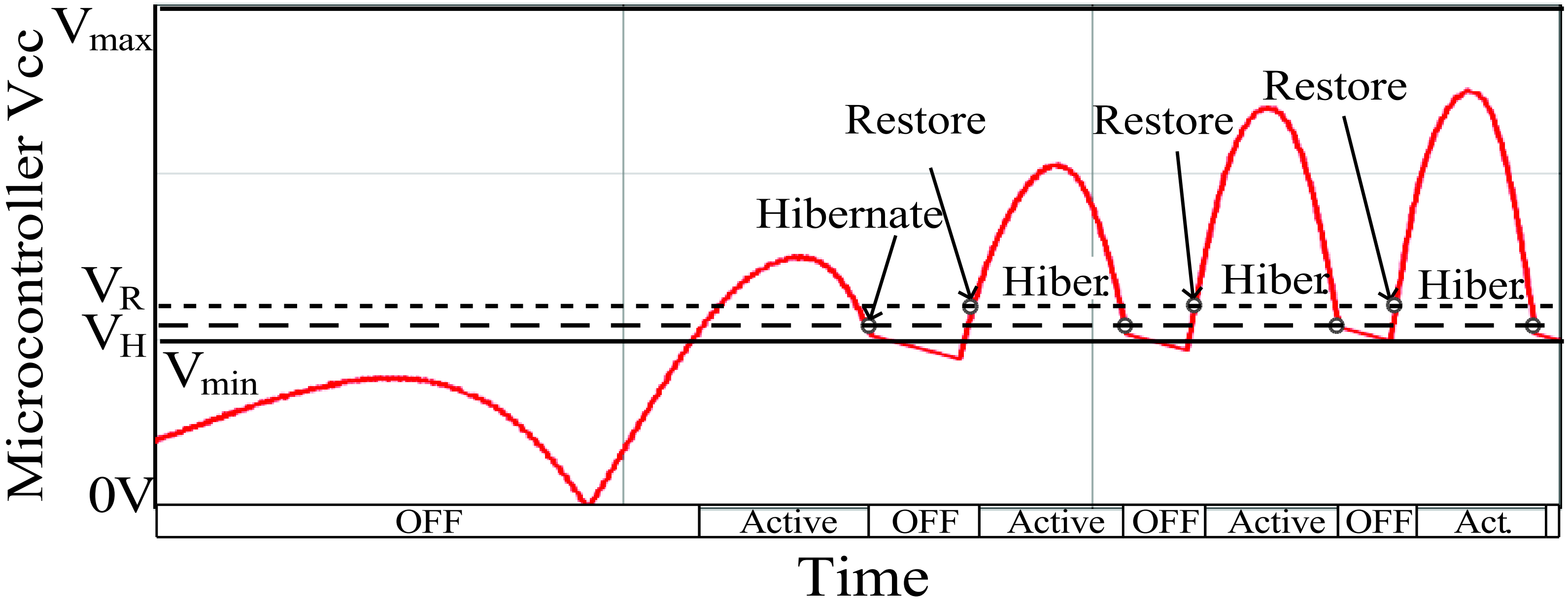
- First/early works in this work
  - Uses concept of check pointing (from fault tolerance)
  - Design/compile-time approach

- Checkpoint placement heuristics, e.g. start of loop/function
  - *Redundant checkpoints/snapshots (causing overheads in both t and E)*
- At a checkpoint, a snapshot is saved if V_{CC} < a threshold
  - *If V_{CC} << threshold, may not be enough time to snapshot*
- After an interruption, restores if a valid snapshot was saved
  - *Code executed since the last checkpoint is lost (therefore is re-executed)*

B. A. Ransford, J. M. Sorber and K. Fu, “Mementos: System Support for Long-Running Computation on RFID-Scale Devices”, *ASPLOS’11*, March 5-11, 2011, Newport Beach, California, USA.

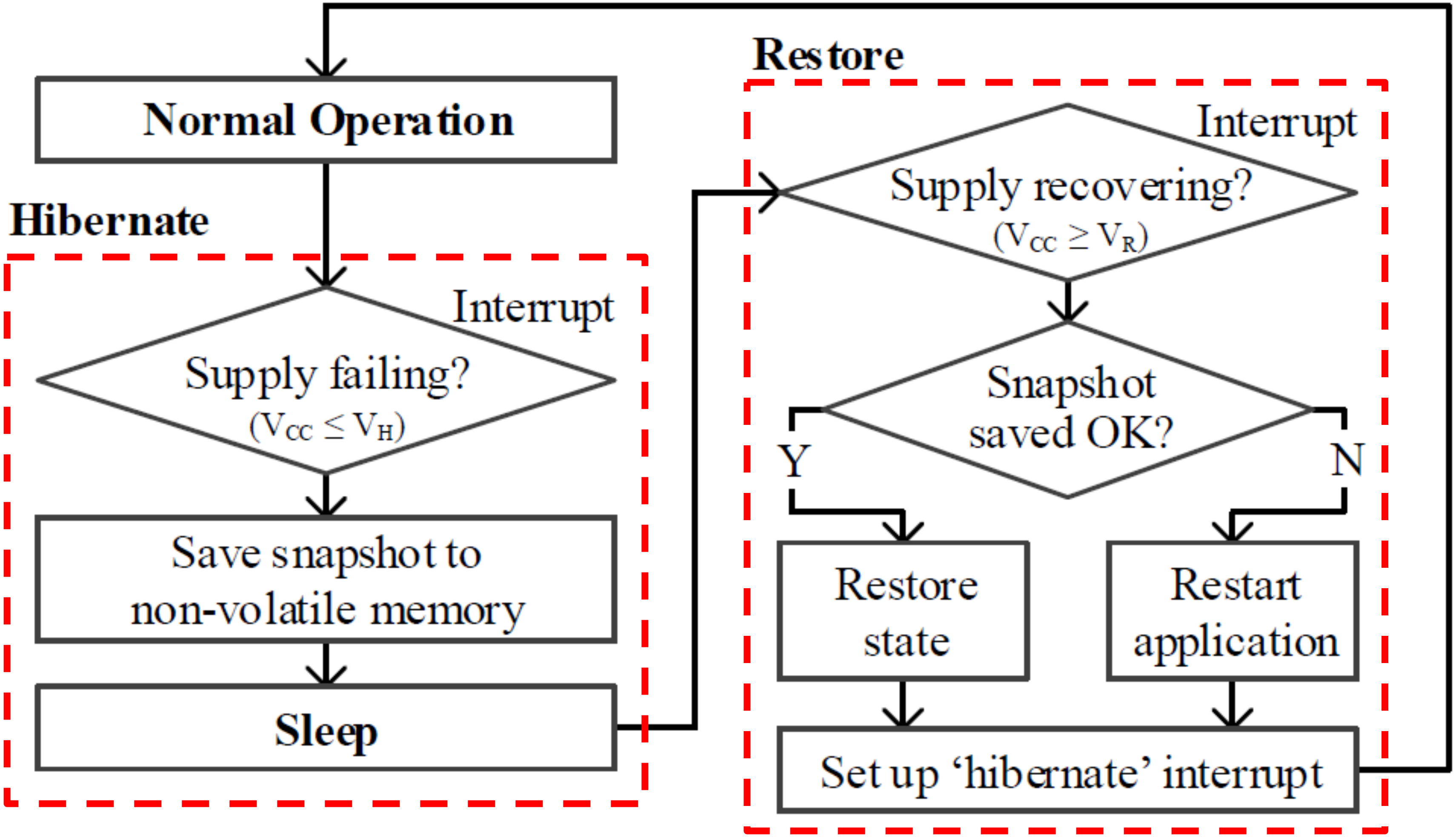
Hibernus

- The motivation for *Hibernus* is to decide when to save a snapshot at run-time
  - Removing overheads (efficiency)
  - Ensuring that a single valid snapshot is always made (reliability)



D. Balsamo, A.S. Weddell, G.V. Merrett, B.M. Al-Hashimi, D. Brunelli, and L. Benini, “Hibernus: Sustaining Computation during Intermittent Supply for Energy-Harvesting Systems,” IEEE Embedded Systems Letters, 2014

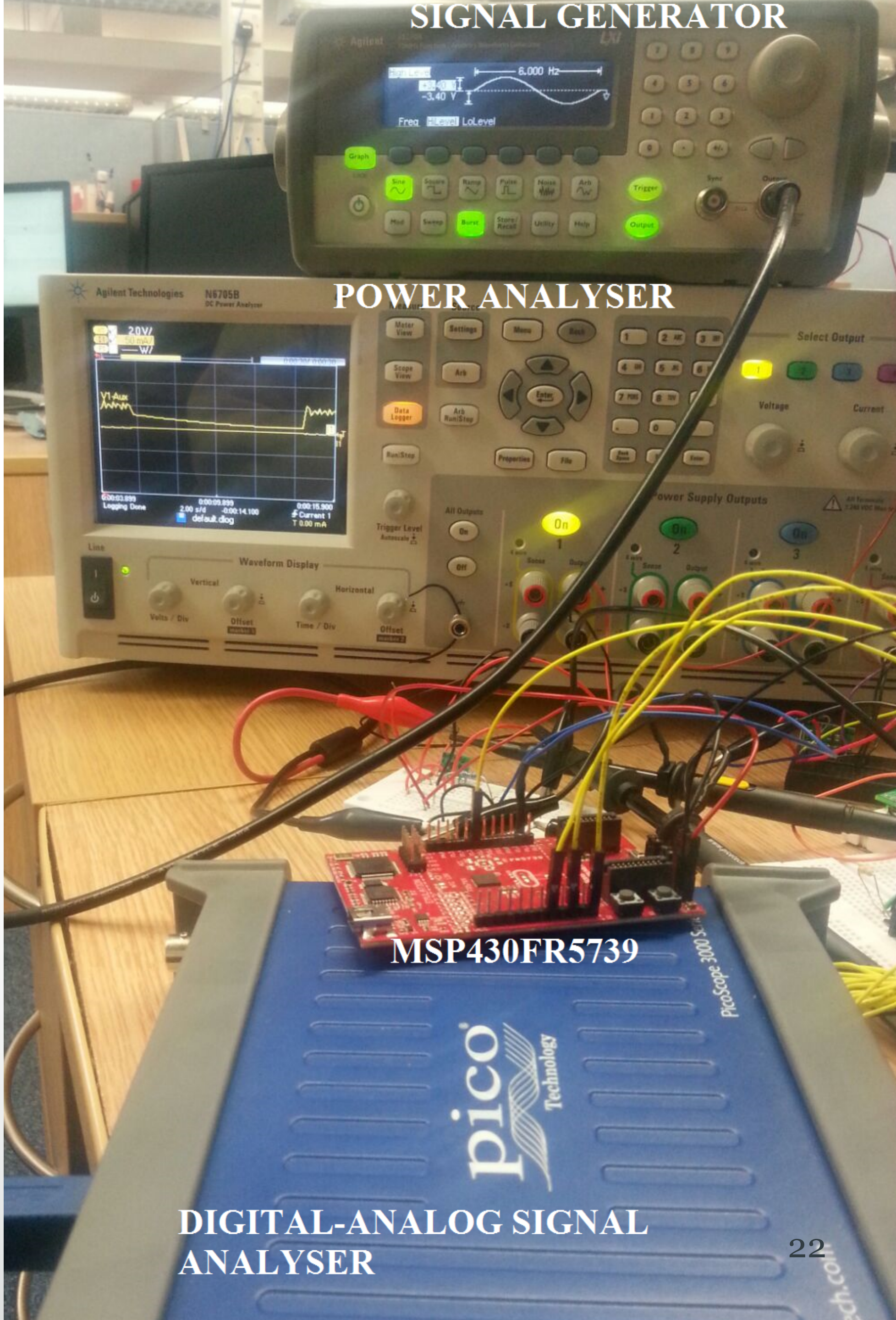
Hibernus: Operation



D. Balsamo, A.S. Weddell, G.V. Merrett, B.M. Al-Hashimi, D. Brunelli, and L. Benini, "Hibernus: Sustaining Computation during Intermittent Supply for Energy-Harvesting Systems," IEEE Embedded Systems Letters, 2014

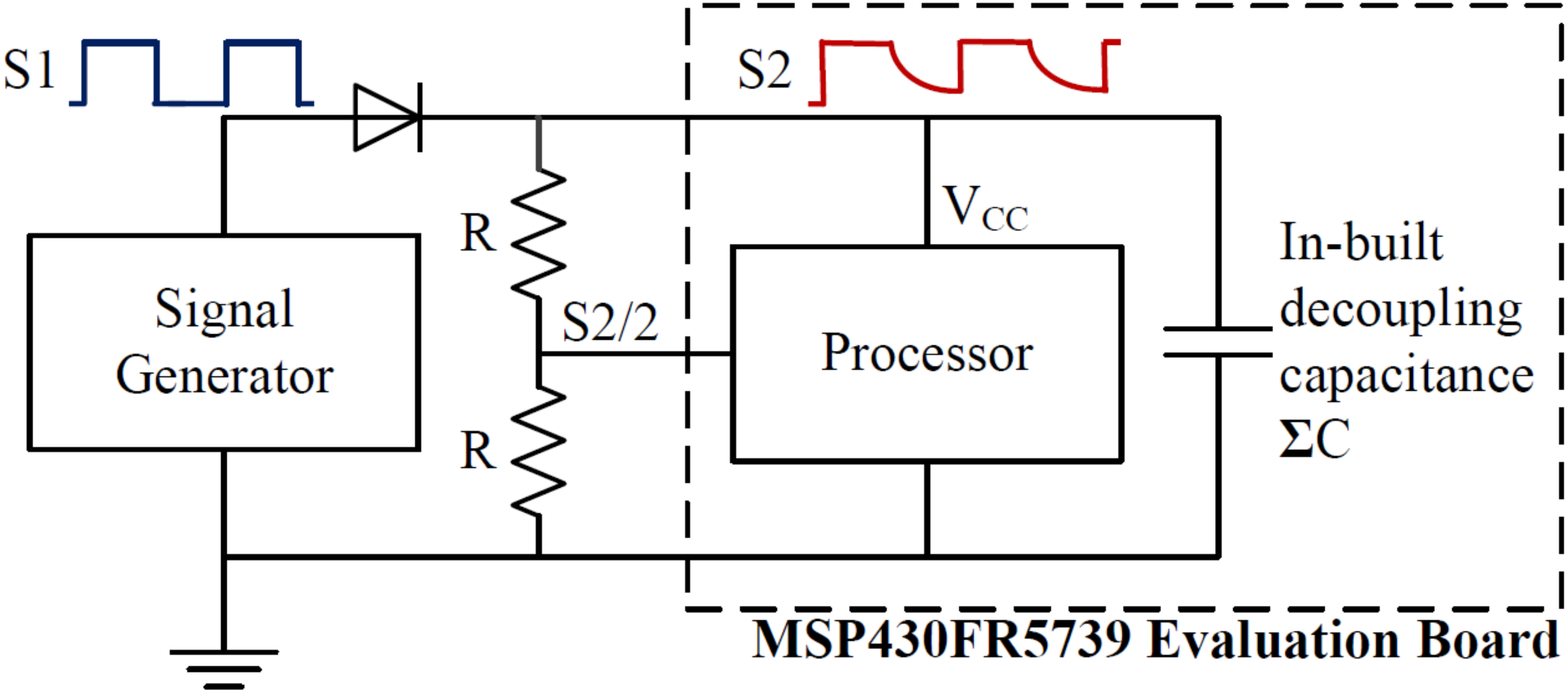
Validation

- Implemented on a TI MSP430-FR5739 microcontroller.
- Tested with
  - Multiple workloads (e.g. FFT)
  - Controlled inputs (signal generator with a range of frequencies DC-20Hz)
  - Real energy harvesters



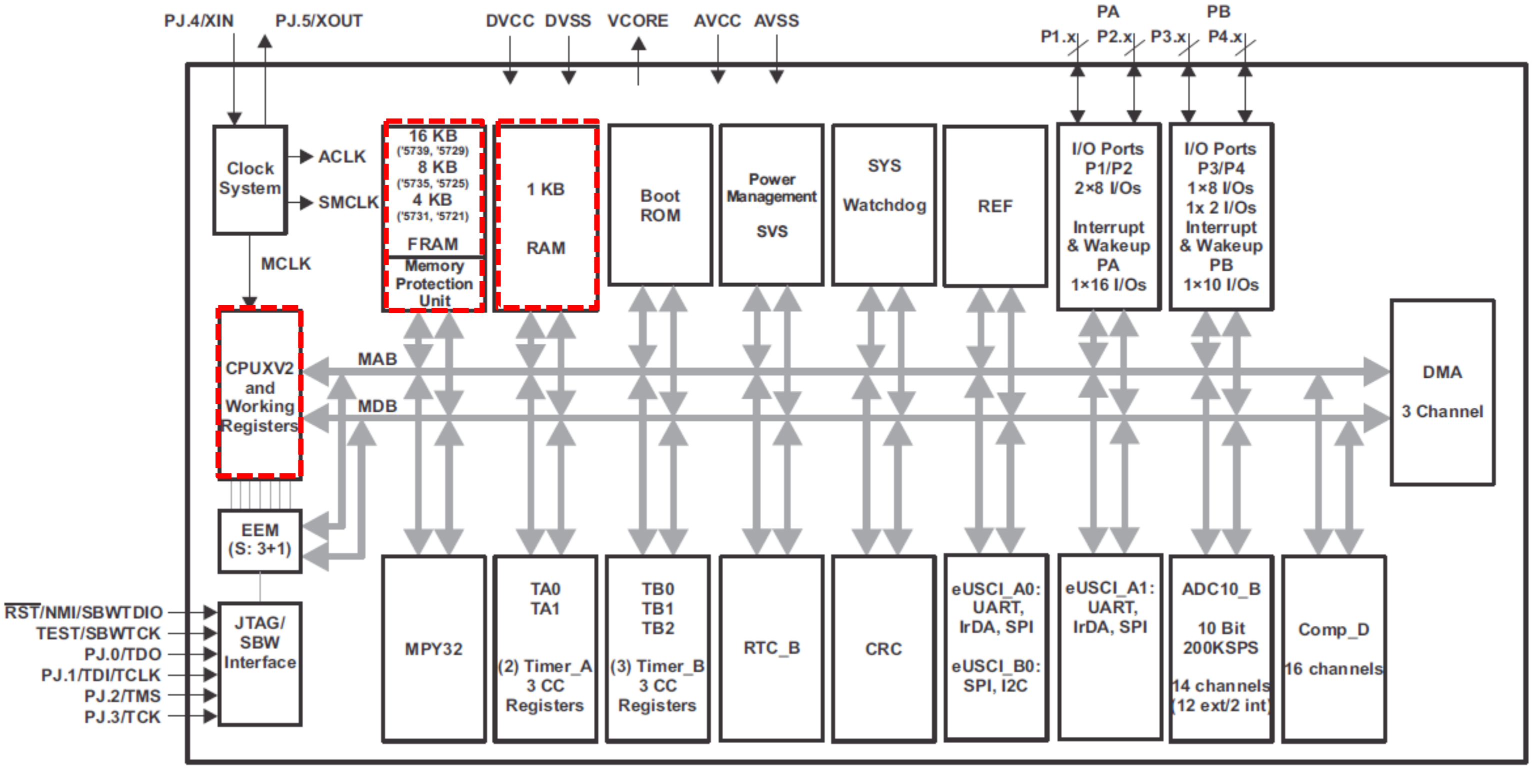
D. Balsamo, A.S. Weddell, G.V. Merrett, B.M. Al-Hashimi, D. Brunelli, and L. Benini, “Hibernus: Sustaining Computation during Intermittent Supply for Energy-Harvesting Systems,” IEEE Embedded Systems Letters, 2014

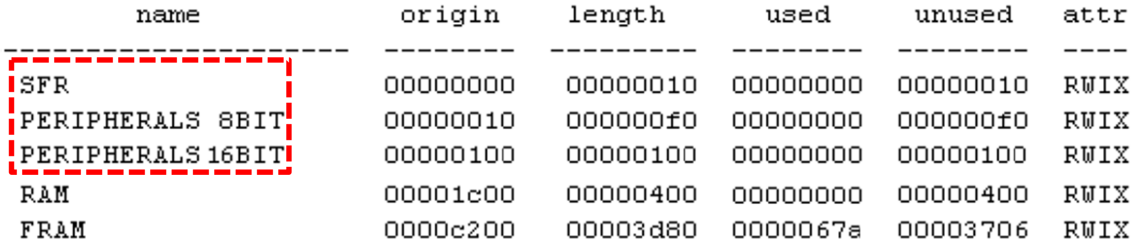
Validation


S1
S2
R
S2/2
R
Vcc
Signal Generator
Processor
In-built decoupling capacitance ΣC
MSP430FR5739 Evaluation Board

D. Balsamo, A.S. Weddell, G.V. Merrett, B.M. Al-Hashimi, D. Brunelli, and L. Benini, “Hibernus: Sustaining Computation during Intermittent Supply for Energy-Harvesting Systems,” IEEE Embedded Systems Letters, 2014

Testbed: TI MSP430FR Block Diagram





```
        name             origin     length     used     unused    attr
---------------------  --------  ---------  --------  --------  ----
SFR                    00000000   00000010  00000000  00000010  RWIX
PERIPHERALS_8BIT       00000010   000000f0  00000000  000000f0  RWIX
PERIPHERALS_16BIT      00000100   00000100  00000000  00000100  RWIX
RAM                    00001c00   00000400  00000000  00000400  RWIX
FRAM                   0000c200   00003d80  0000067a  00003706  RWIX
```

The total time needed to store all Core Registers, all RAM and all General Registers is only 1.3ms.

Hibernation Threshold

- Energy consumed hibernating E_σ depends on the size of the volatile memory and the energy for each byte.

$$E_\sigma = n_\alpha E_\alpha + n_\beta E_\beta$$

- Given the total capacitance ($\sum C$), the energy stored E_δ between a given voltage V_H and V_{min} is:

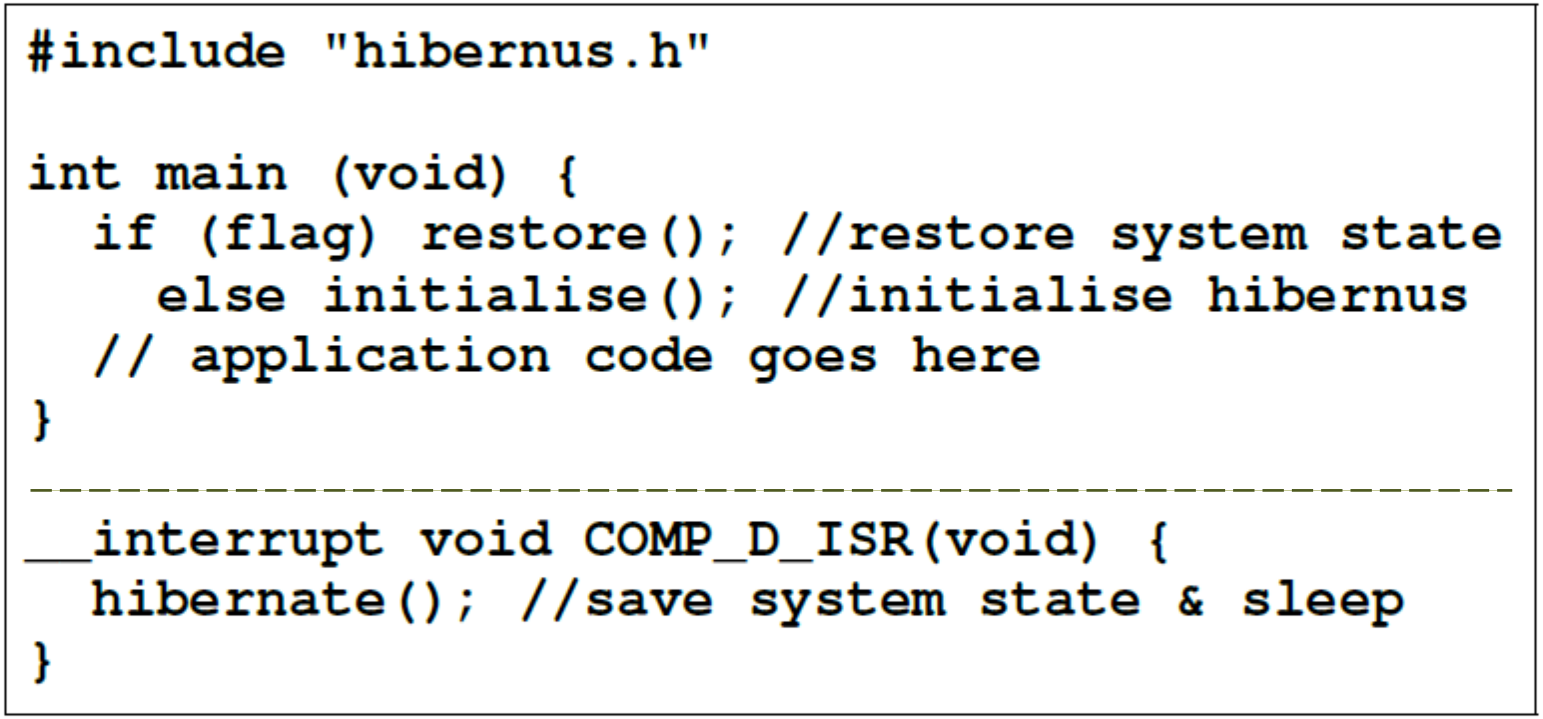
$$E_\delta = \frac{V_H^2 - V_{min}^2}{2} \sum C$$

- To ensure stability, V_H is set to that $E_\sigma < E_\delta$, to enable complete hibernation.

D. Balsamo, A.S. Weddell, G.V. Merrett, B.M. Al-Hashimi, D. Brunelli, and L. Benini, "Hibernus: Sustaining Computation during Intermittent Supply for Energy-Harvesting Systems," IEEE Embedded Systems Letters, 2014

The *Hibernus* Library

- *Hibernus* functionality is contained within the `hibernus.h` library file;
- Application developers only need to include this library the `initialise()`, `hibernate()` and `restore()` routine;



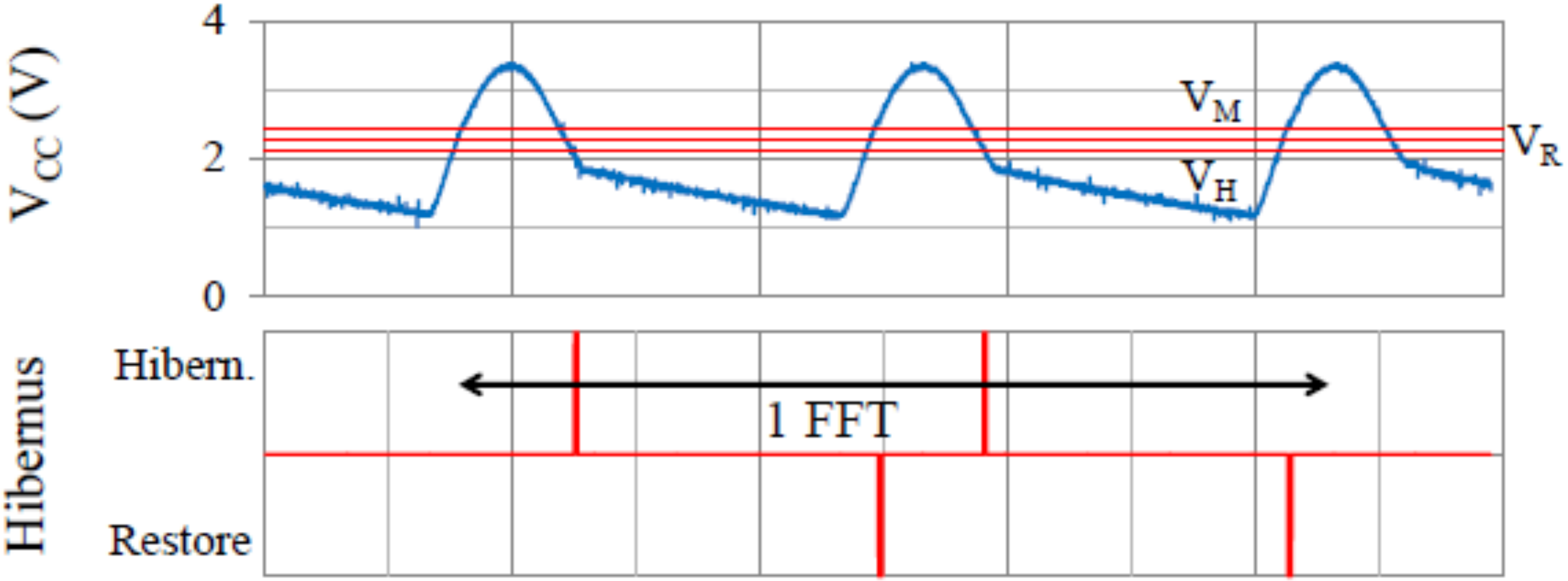
```
#include "hibernus.h"

int main (void) {
  if (flag) restore(); //restore system state
    else initialise(); //initialise hibernus
  // application code goes here
}
---------------------------------------------
__interrupt void COMP_D_ISR(void) {
  hibernate(); //save system state & sleep
}
```

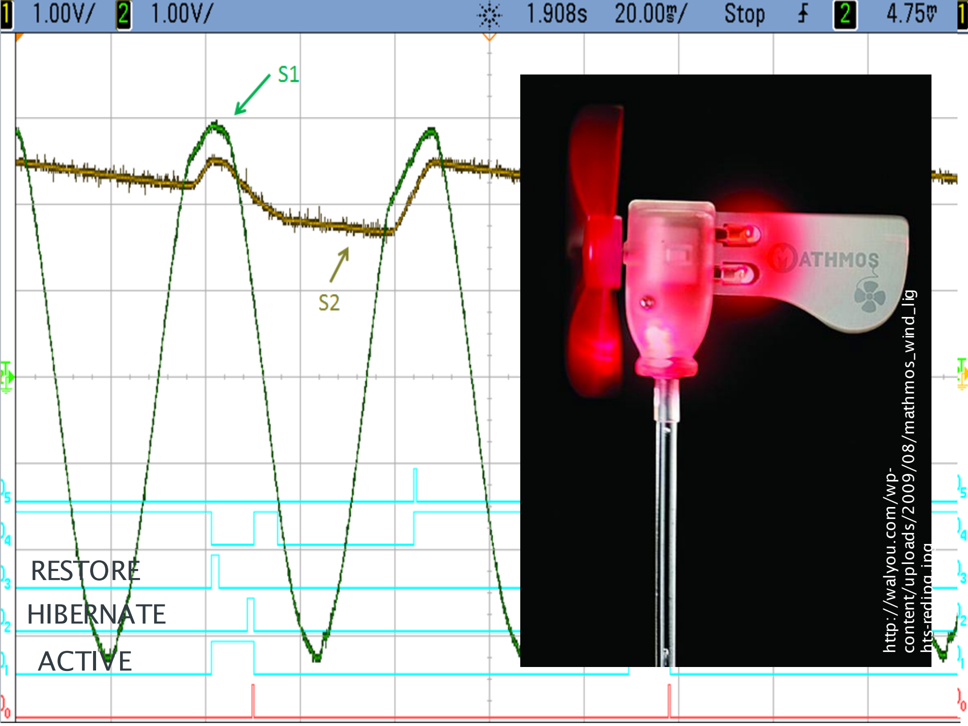
D. Balsamo, A.S. Weddell, G.V. Merrett, B.M. Al-Hashimi, D. Brunelli, and L. Benini, “Hibernus: Sustaining Computation during Intermittent Supply for Energy-Harvesting Systems,” IEEE Embedded Systems Letters, 2014

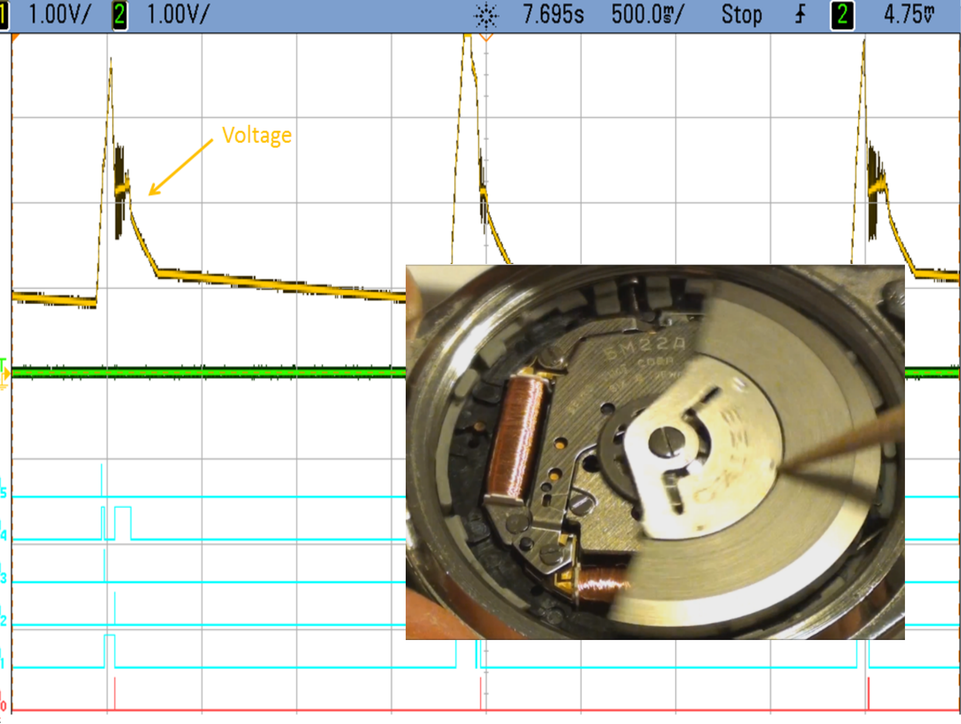
Hibernus Results

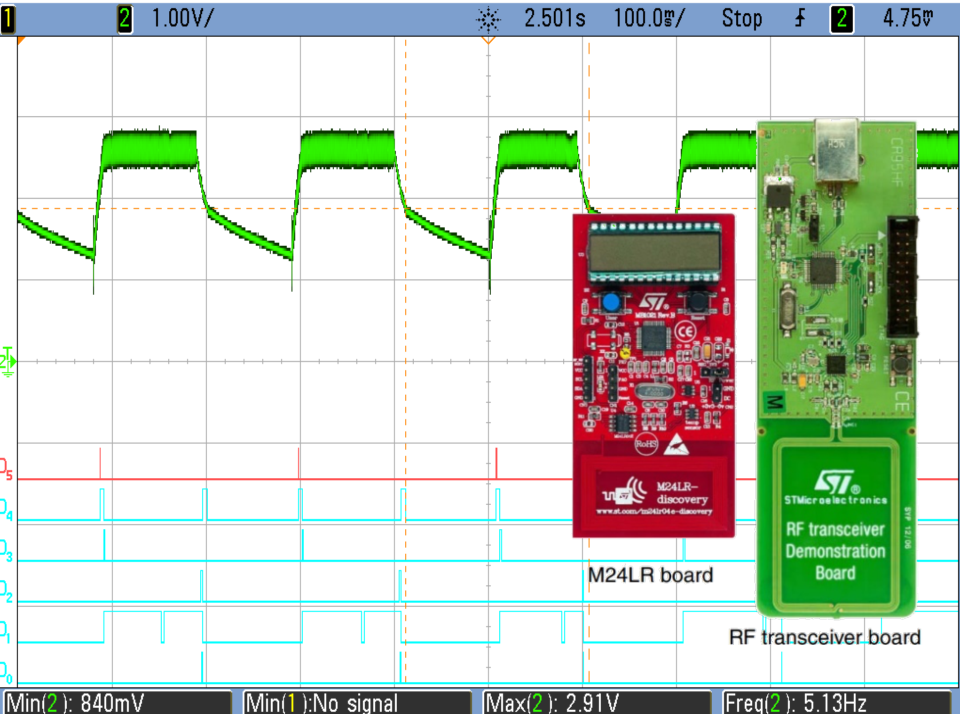
- FFT of tri-axial accelerometer data

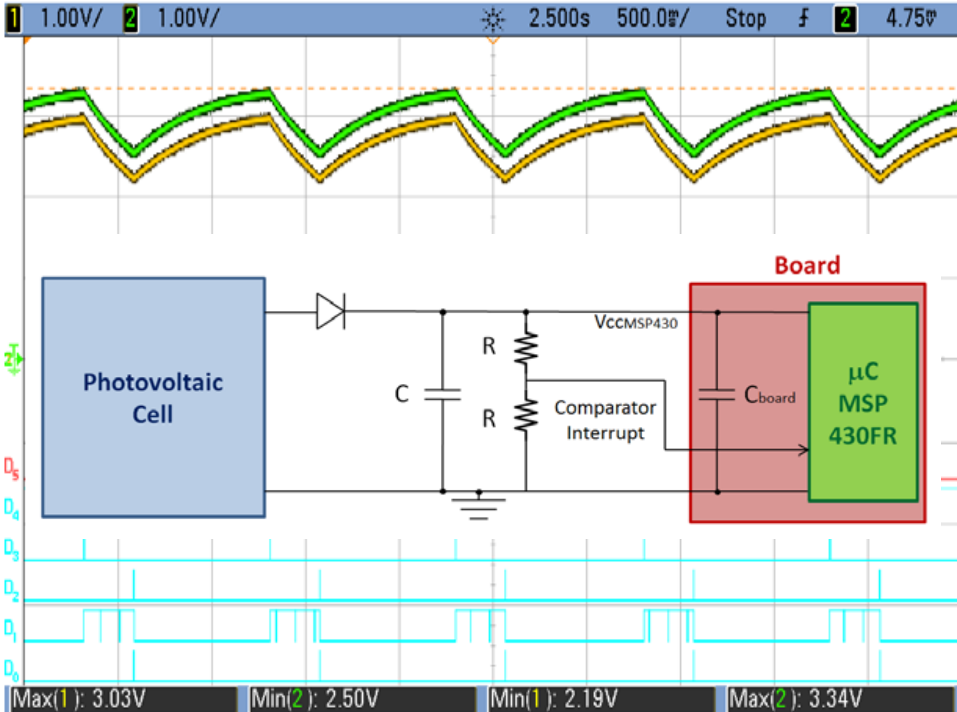


D. Balsamo, A.S. Weddell, G.V. Merrett, B.M. Al-Hashimi, D. Brunelli, and L. Benini, "Hibernus: Sustaining Computation during Intermittent Supply for Energy-Harvesting Systems," IEEE Embedded Systems Letters, 2014


1 1.00V/ 2 1.00V/ 1.908s 20.00ms/ Stop 2 4.75V
S1
S2
RESTORE
HIBERNATE
ACTIVE
ATHMOS
http://walyou.com/wp-content/uploads/2009/08/mathmos_wind_lights-reding.jpg

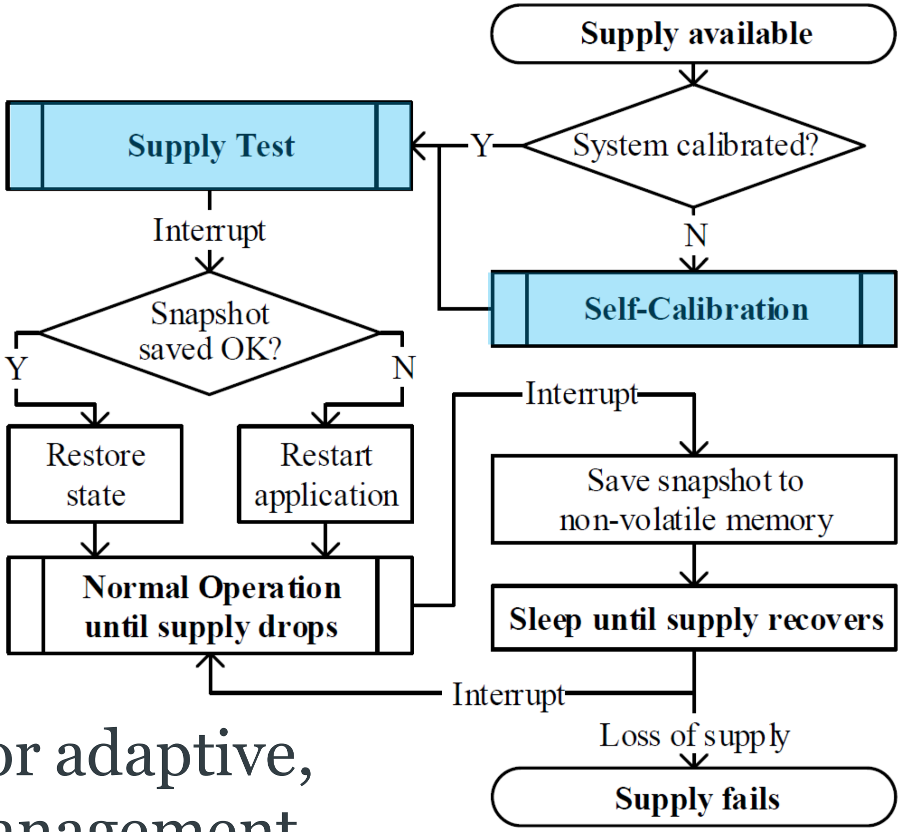

1 1.00V/ 2 1.00V/ 7.695s 500.0ms/ Stop 2 4.75V
Voltage


1 2 1.00V/ 2.501s 100.0ms/ Stop 2 4.75V
M24LR board
RF transceiver board
STMicroelectronics RF transceiver Demonstration Board
M24LR-discovery
Min(2): 840mV
Min(1):No signal
Max(2): 2.91V
Freq(2): 5.13Hz


1 1.00V/ 2 1.00V/ 2.500s 500.0ms/ Stop 2 4.75V
Board
Photovoltaic Cell
C
R
R
VccMSP430
Comparator Interrupt
Cboard
µC MSP 430FR
Max(1): 3.03V
Min(2): 2.50V
Min(1): 2.19V
Max(2): 3.34V

Automatic Calibration of Hibernus

- Hibernus required manual design-time calibration:
  - Select hibernate threshold based on the worst case *[Source Dependent]*
  - Select hibernate threshold based on *C'* *[Platform Dependent]*
  - Select restore threshold based on source dynamics *[Source Dependent]*

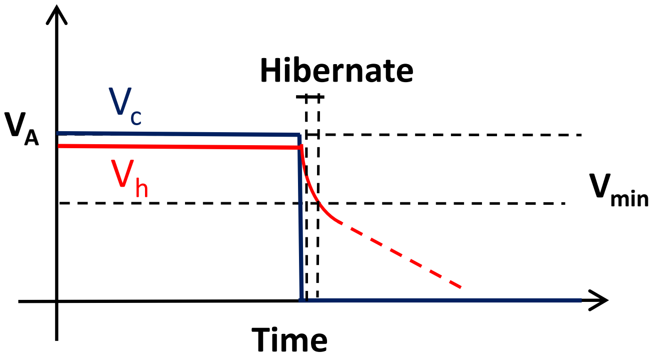


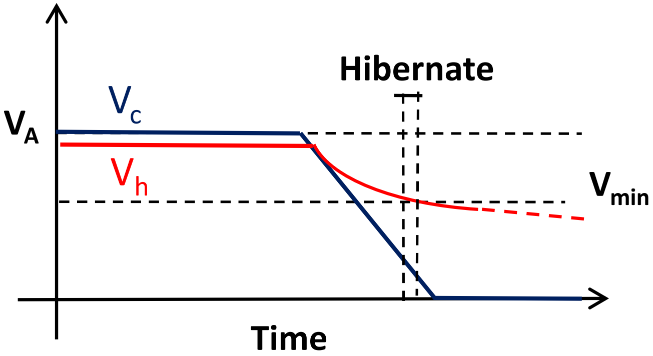
- Hibernus++ adds process for adaptive, run-time calibration and management

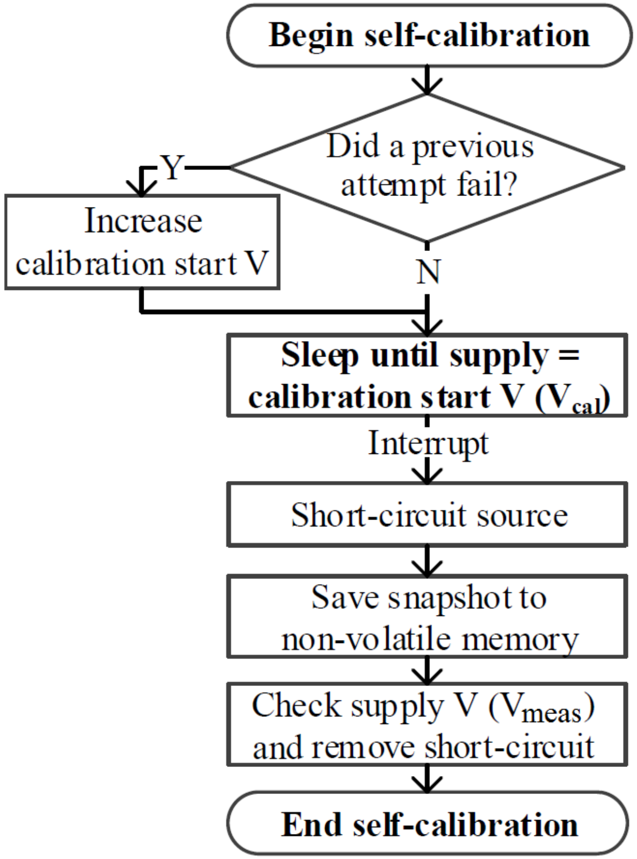
D. Balsamo, A.S. Weddell, A. Das, A. Rodriguez Arreola, D. Brunelli, B.M. Al-Hashimi, G.V. Merrett, L. Benini, (2016) Hibernus++: a self-calibrating and adaptive system for transiently-powered embedded devices. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 1-13.

Hibernus++ Adaptive Hibernation

- *Calibration* Process







Begin self-calibration

Did a previous attempt fail?

Y

Increase calibration start V

N

Sleep until supply = calibration start V (V_{cal})

Interrupt

Short-circuit source

Save snapshot to non-volatile memory

Check supply V (V_{meas}) and remove short-circuit

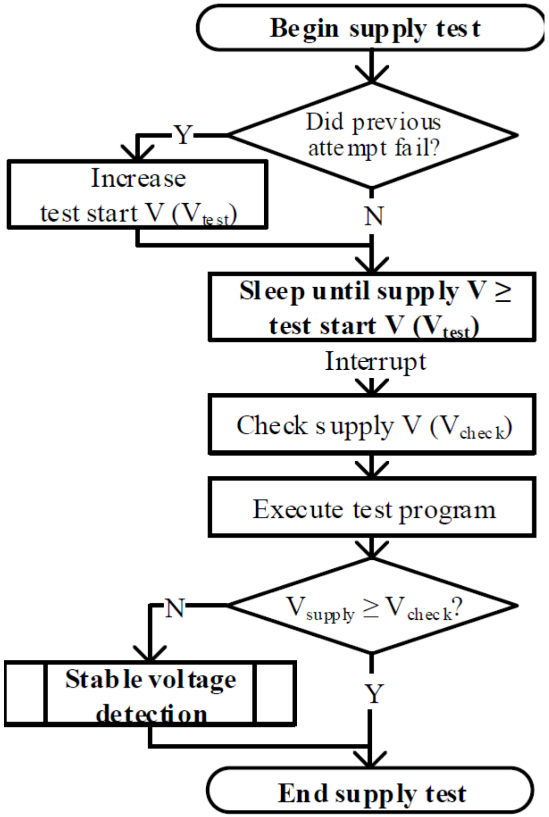
End self-calibration

Still pick the 'worst case' power loss (so that can always hibernate), put characterize the platform

D. Balsamo, A.S. Weddell, A. Das, A. Rodriguez Arreola, D. Brunelli, B.M. Al-Hashimi, G.V. Merrett, L. Benini, (2016) Hibernus++: a self-calibrating and adaptive system for transiently-powered embedded devices. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 1-13.

Hibernus++ Adaptive Restore

- *Source Classification* Process



Begin supply test

Did previous attempt fail?

Y

Increase test start V (V_{test})

N

Sleep until supply V ≥ test start V (V_{test})

Interrupt

Check supply V (V_{check})

Execute test program

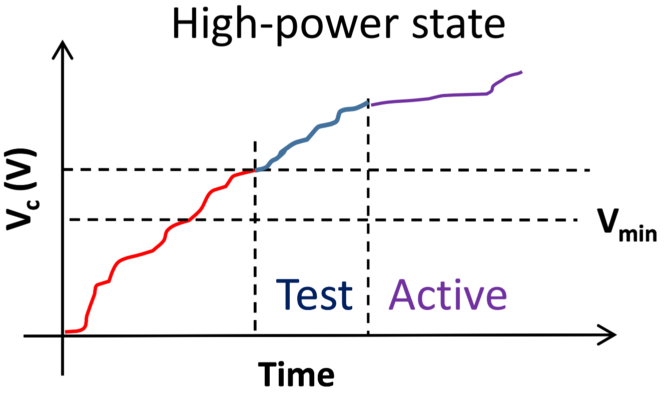
$V_{supply} \geq V_{check}$?

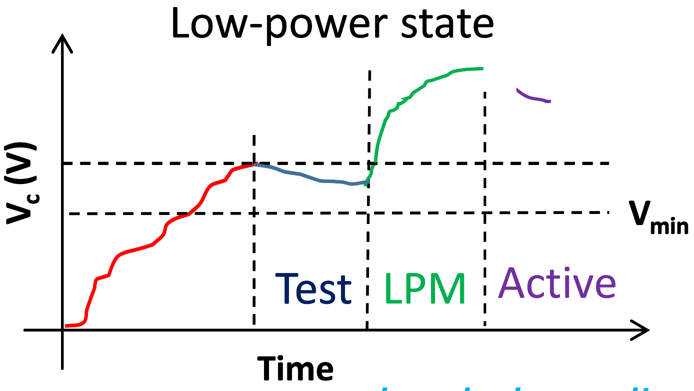
N

Stable voltage detection

Y

End supply test

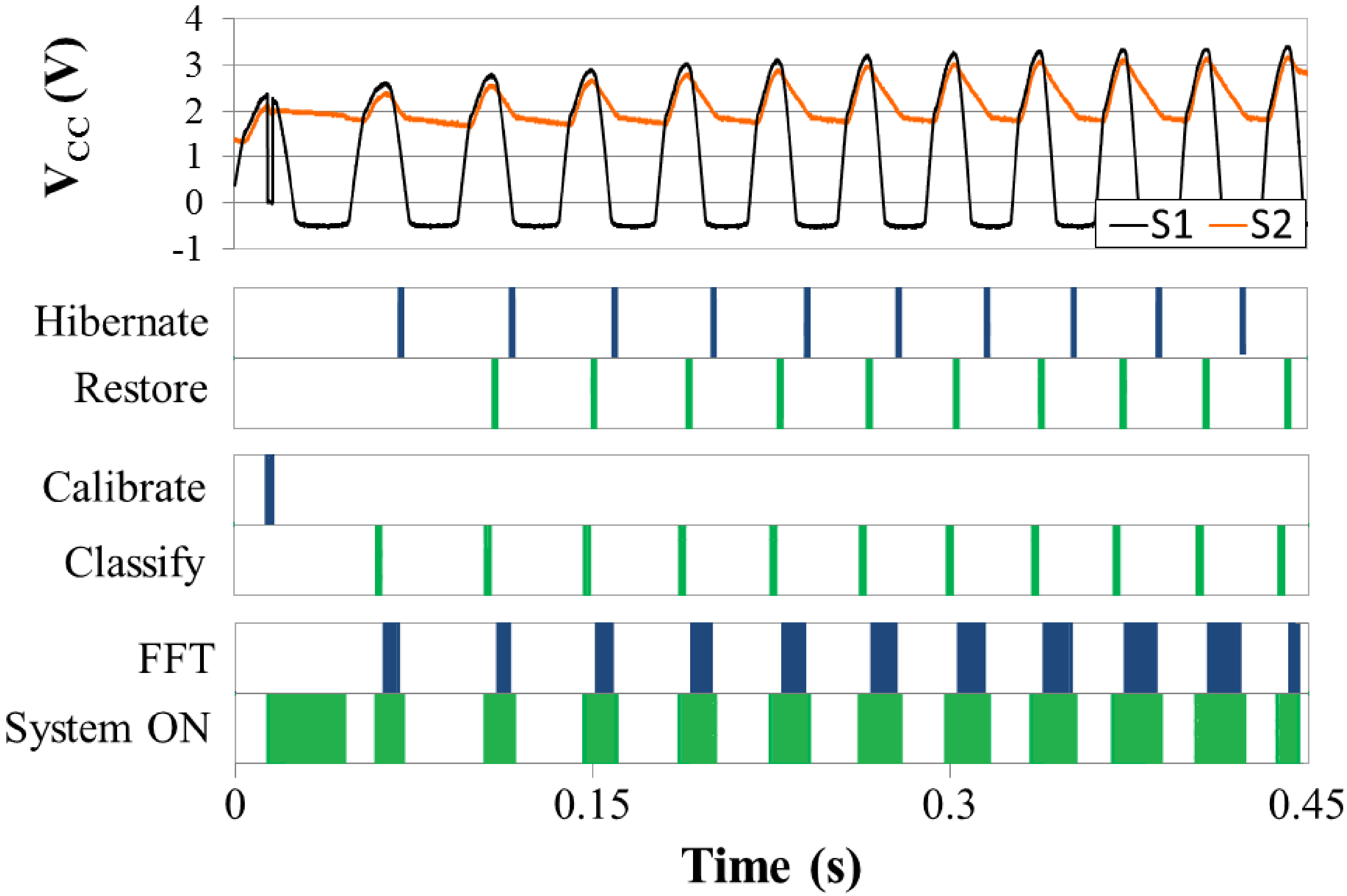




Characterize the source dynamics each time we want to restore, to select the best policy

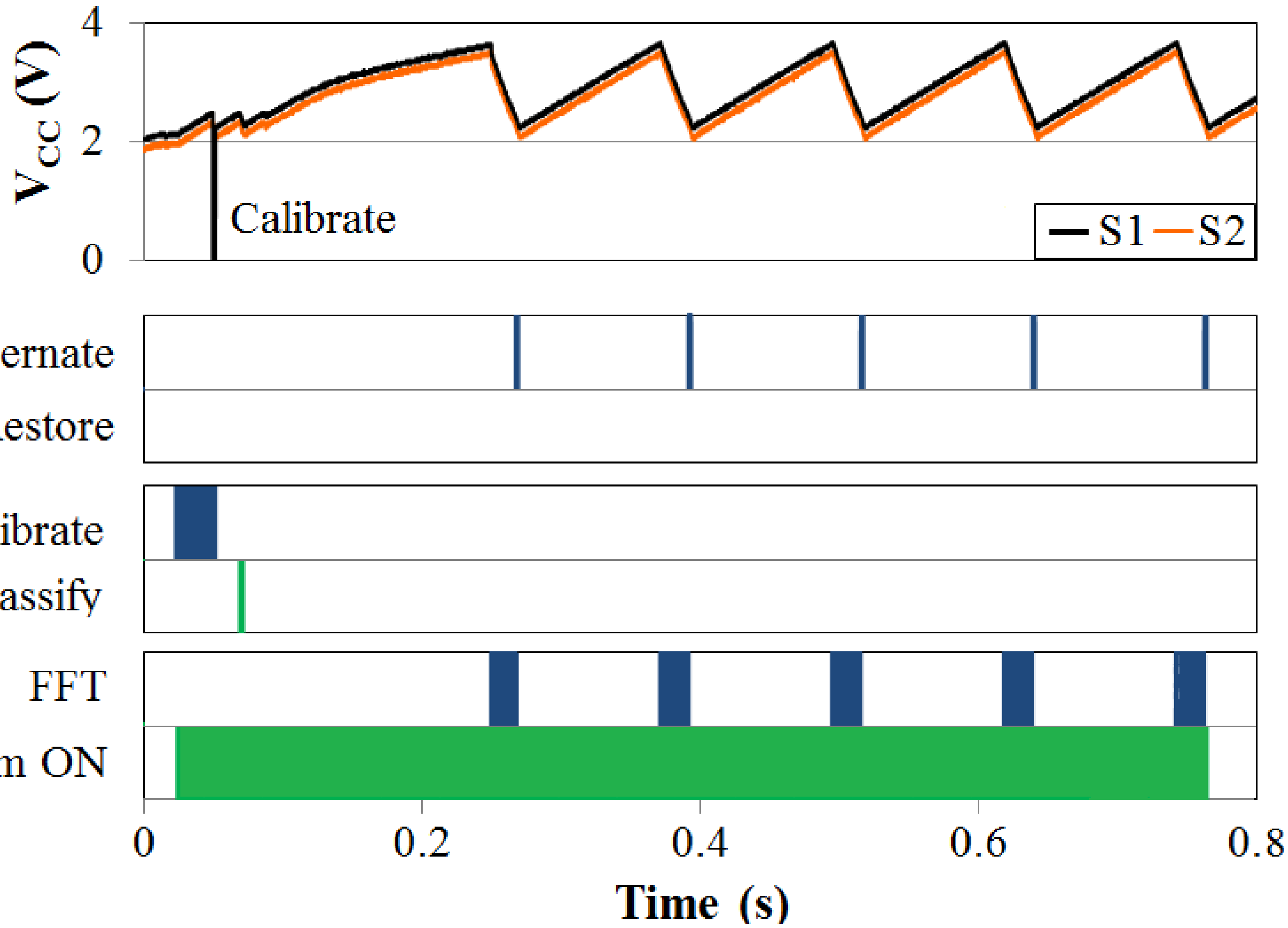
D. Balsamo, A.S. Weddell, A. Das, A. Rodriguez Arreola, D. Brunelli, B.M. Al-Hashimi, G.V. Merrett, L. Benini, (2016) Hibernus++: a self-calibrating and adaptive system for transiently-powered embedded devices. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 1-13.

Hibernus++ Results



D. Balsamo, A.S. Weddell, A. Das, A. Rodriguez Arreola, D. Brunelli, B.M. Al-Hashimi, G.V. Merrett, L. Benini, (2016) Hibernus++: a self-calibrating and adaptive system for transiently-powered embedded devices. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 1-13.

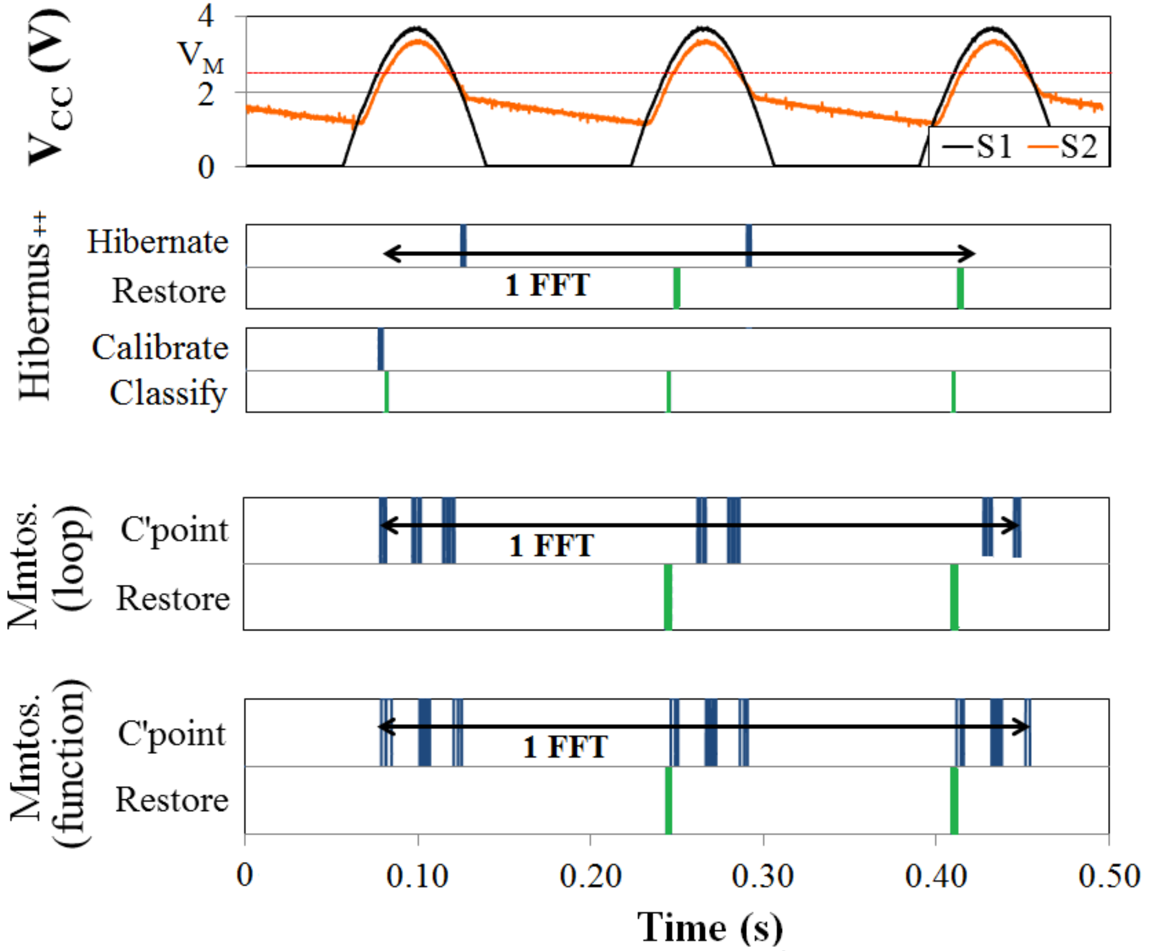
Hibernus++ Results



Also go to sleep after hibernating, until restore policy is met again

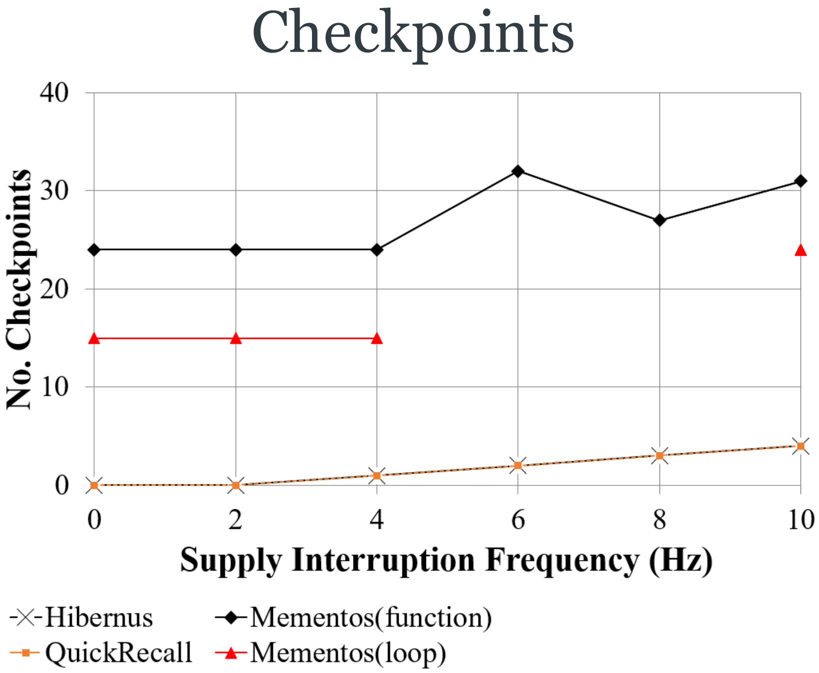
D. Balsamo, A.S. Weddell, A. Das, A. Rodriguez Arreola, D. Brunelli, B.M. Al-Hashimi, G.V. Merrett, L. Benini, (2016) Hibernus++: a self-calibrating and adaptive system for transiently-powered embedded devices. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 1-13.

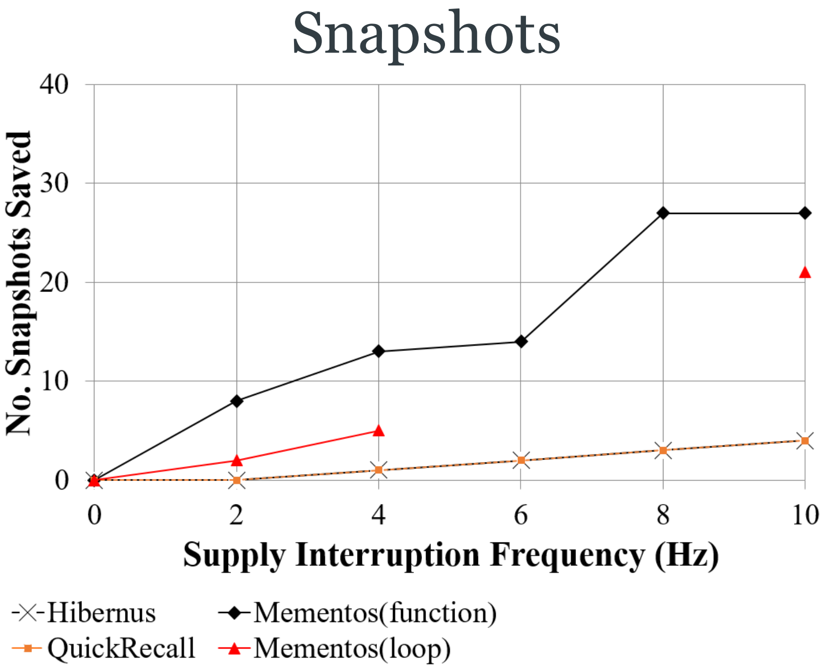
Comparison



D. Balsamo, A.S. Weddell, A. Das, A. Rodriguez Arreola, D. Brunelli, B.M. Al-Hashimi, G.V. Merrett, L. Benini, (2016) Hibernus++: a self-calibrating and adaptive system for transiently-powered embedded devices. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 1-13.

Comparison



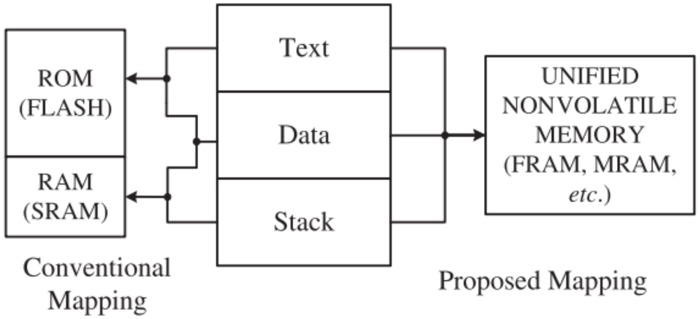


- **Hibernus**: checkpoints (and snapshots) every time the voltage drops (i.e. supply interrupted)
- **Mementos**: checkpoints depending on design-/compile-time placement heuristics, and snapshots if a checkpoint occurs AND $V_{CC} < V_{min}$

A. Rodriguez, D. Balsamo, A. Das, A.S. Weddell, D. Brunelli, B.M. Al-Hashimi, G.V. Merrett, (2015) Approaches to Transient Computing for Energy Harvesting Systems: A Quantitative Evaluation. In, *Int'l Workshop Energy Harvesting and Energy Neutral Sensing Systems (ENSsys) 2015, Korea, Republic of, 01 Nov 2015.*

QuickRecall

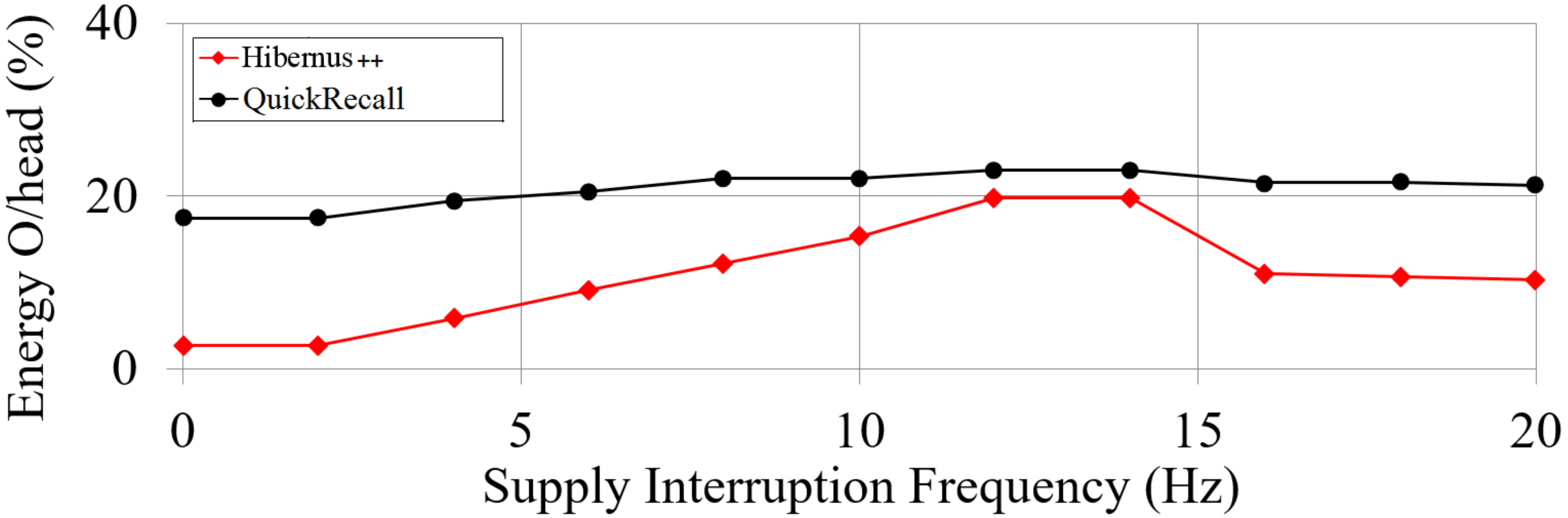
- Takes advantage of a unified memory system
  - Data + program memory is always in NVM



- Therefore, a snapshot only transfer registers to NVM
  - Elegant solution (+ likewise with NVPs)
  - Much quicker
  - *NVM typically consumes more power…*

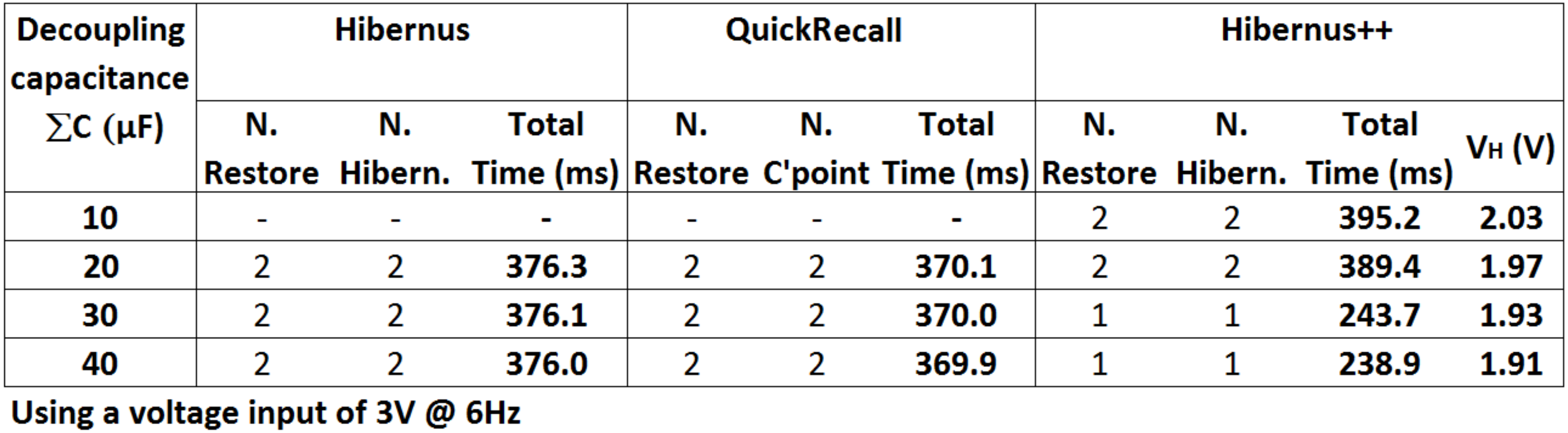
Hrishikesh Jayakumar, Arnab Raha, Woo Suk Lee, and Vijay Raghunathan. 2015. QuickRecall: A HW/SW Approach for Computing across Power Cycles in Transiently Powered Computers. *J. Emerg. Technol. Comput. Syst.* 12, 1, Article 8 (August 2015), 19 pages.

Hibernus++ vs Quickrecall



D. Balsamo, A.S. Weddell, A. Das, A. Rodriguez Arreola, D. Brunelli, B.M. Al-Hashimi, G.V. Merrett, L. Benini, (2016) Hibernus++: a self-calibrating and adaptive system for transiently-powered embedded devices. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 1-13.

Benefits of Hibernus++



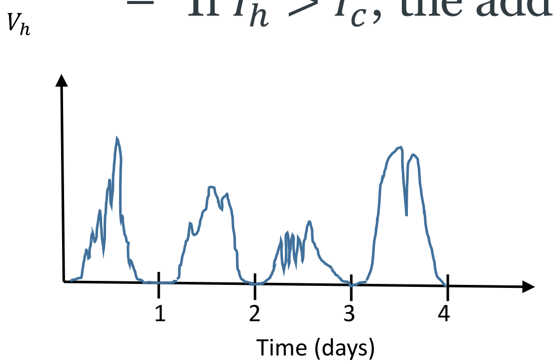
Decoupling capacitance $\sum C$ (μF)	Hibernus			QuickRecall			Hibernus++			
	N. Restore	N. Hibern.	Total Time (ms)	N. Restore	N. C'point	Total Time (ms)	N. Restore	N. Hibern.	Total Time (ms)	V_H (V)
10	-	-	**-**	-	-	**-**	2	2	**395.2**	**2.03**
20	2	2	**376.3**	2	2	**370.1**	2	2	**389.4**	**1.97**
30	2	2	**376.1**	2	2	**370.0**	1	1	**243.7**	**1.93**
40	2	2	**376.0**	2	2	**369.9**	1	1	**238.9**	**1.91**

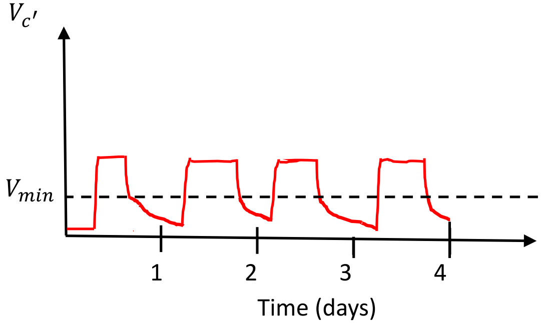
Using a voltage input of 3V @ 6Hz

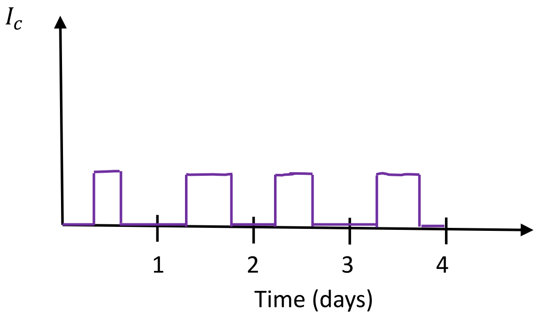
D. Balsamo, A.S. Weddell, A. Das, A. Rodriguez Arreola, D. Brunelli, B.M. Al-Hashimi, G.V. Merrett, L. Benini, (2016) Hibernus++: a self-calibrating and adaptive system for transiently-powered embedded devices. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 1-13.

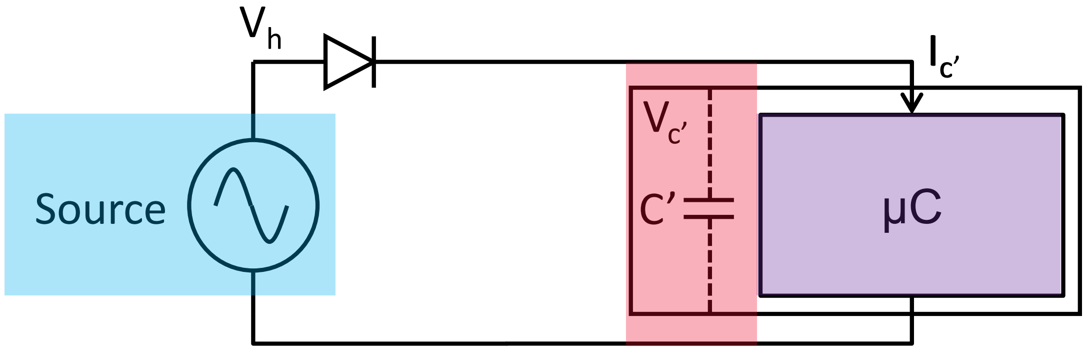
Transient Computing

- Hibernus (and other techniques) are *on* or *off*
  - If $I_h < I_c$, the system hibernates
  - If $I_h > I_c$, the additional power is typically wasted









- Use adaptive operation, like energy-neutral computing?

Outline

towards storage-less and power-centric systems

Energy-Harvesting Systems

Operating from harvested energy

Energy-Neutral Computing

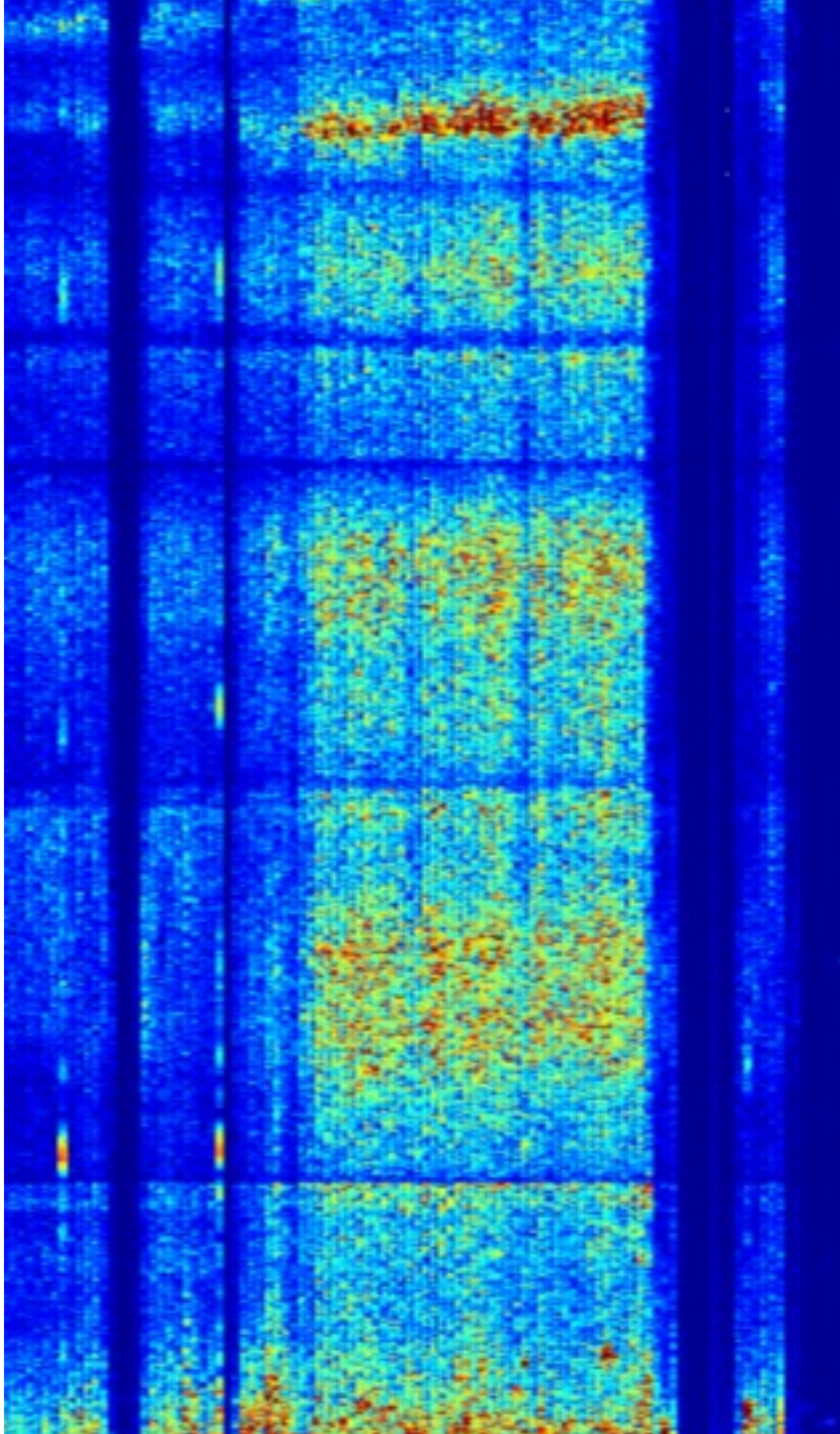
Buffering energy for a battery-like supply

Transient Computing

Computation when power is available

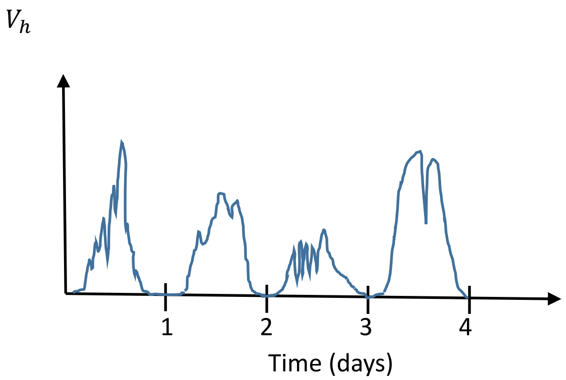
Power-Neutral Computing

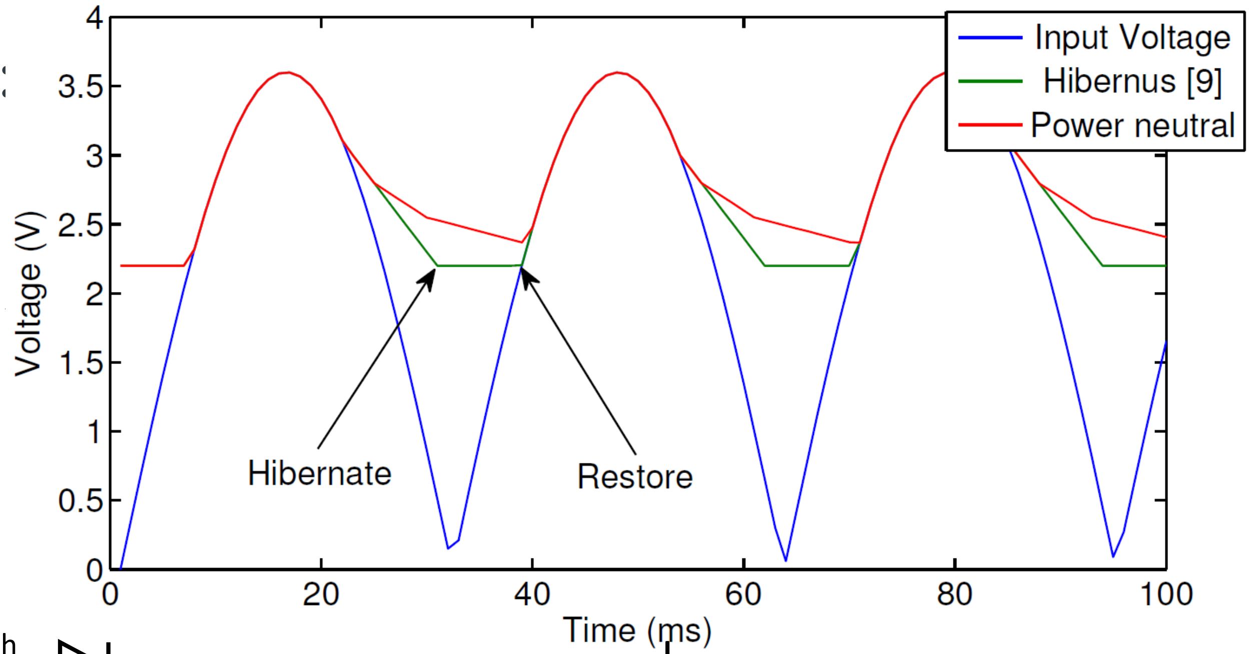
Adaptive computation when power is available

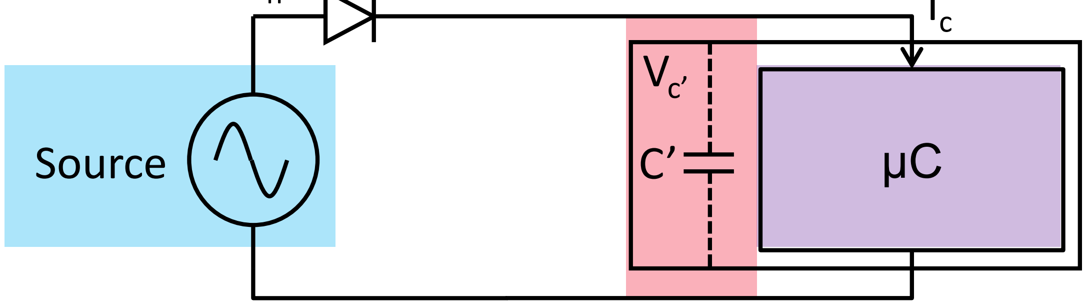


Power-Neutral Computing

- Energy Neutrality:
- Power Neutrality:



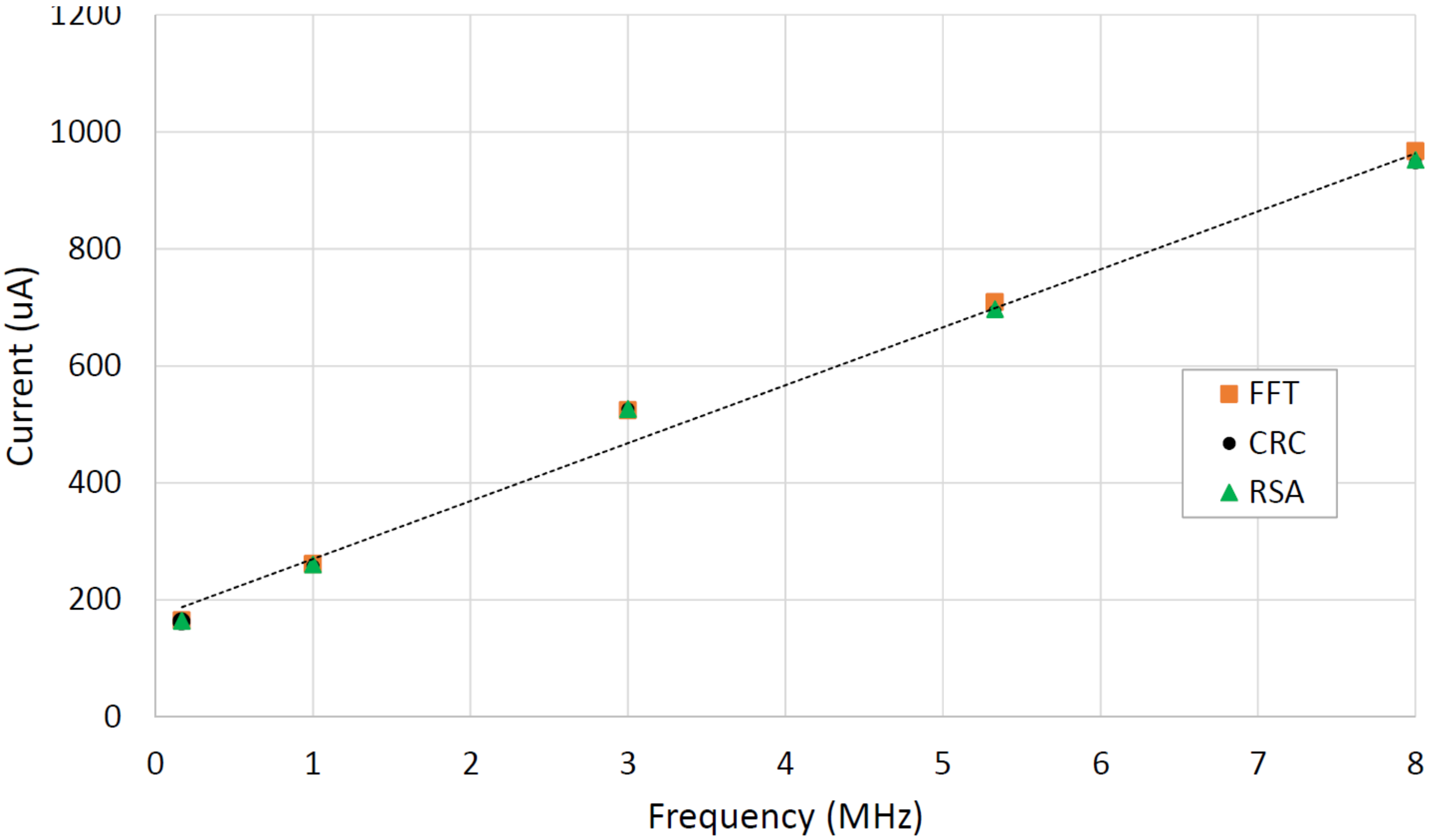




$$\lim_{C'\to 0} \frac{P_h(t)}{P_c(t)} = 1$$ or, as C' tends to zero, $P_{harvest}(t) = P_{consume}(t)$

Dynamic Frequency Scaling (DFS)

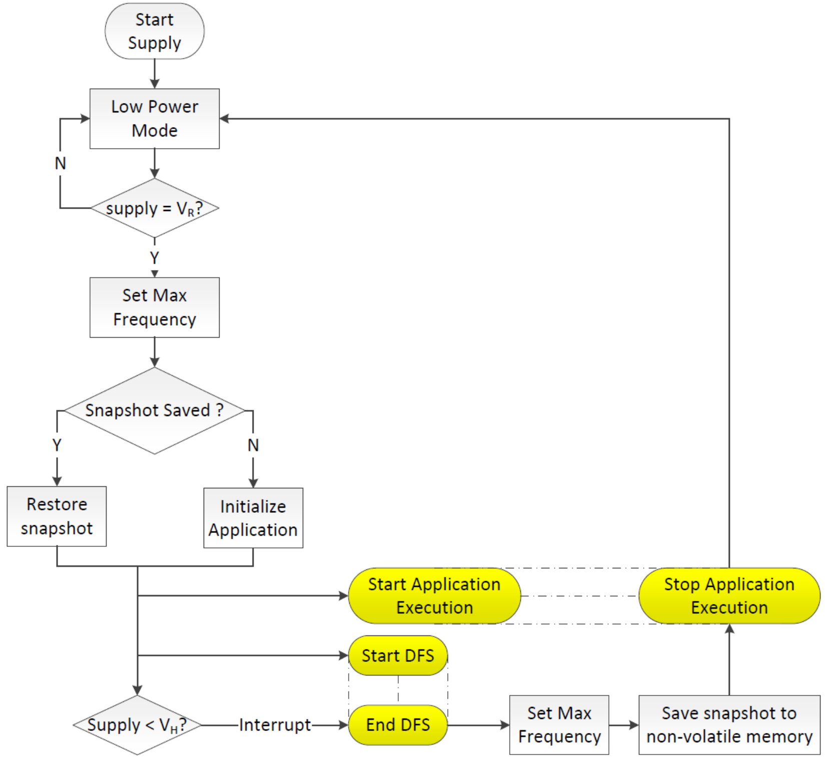
- On an MSP430

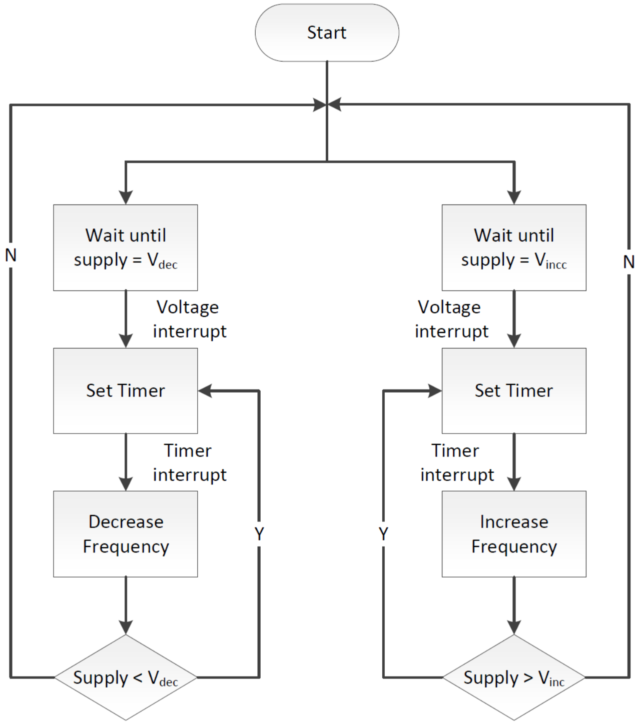


D. Balsamo, A. Das, A.S. Weddell, D. Brunelli, B.M. Al-Hashimi, G.V. Merrett, L. Benini, (2016) Graceful Performance Modulation for Power-Neutral Transient Computing Systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*

Hibernus with Graceful Degradation

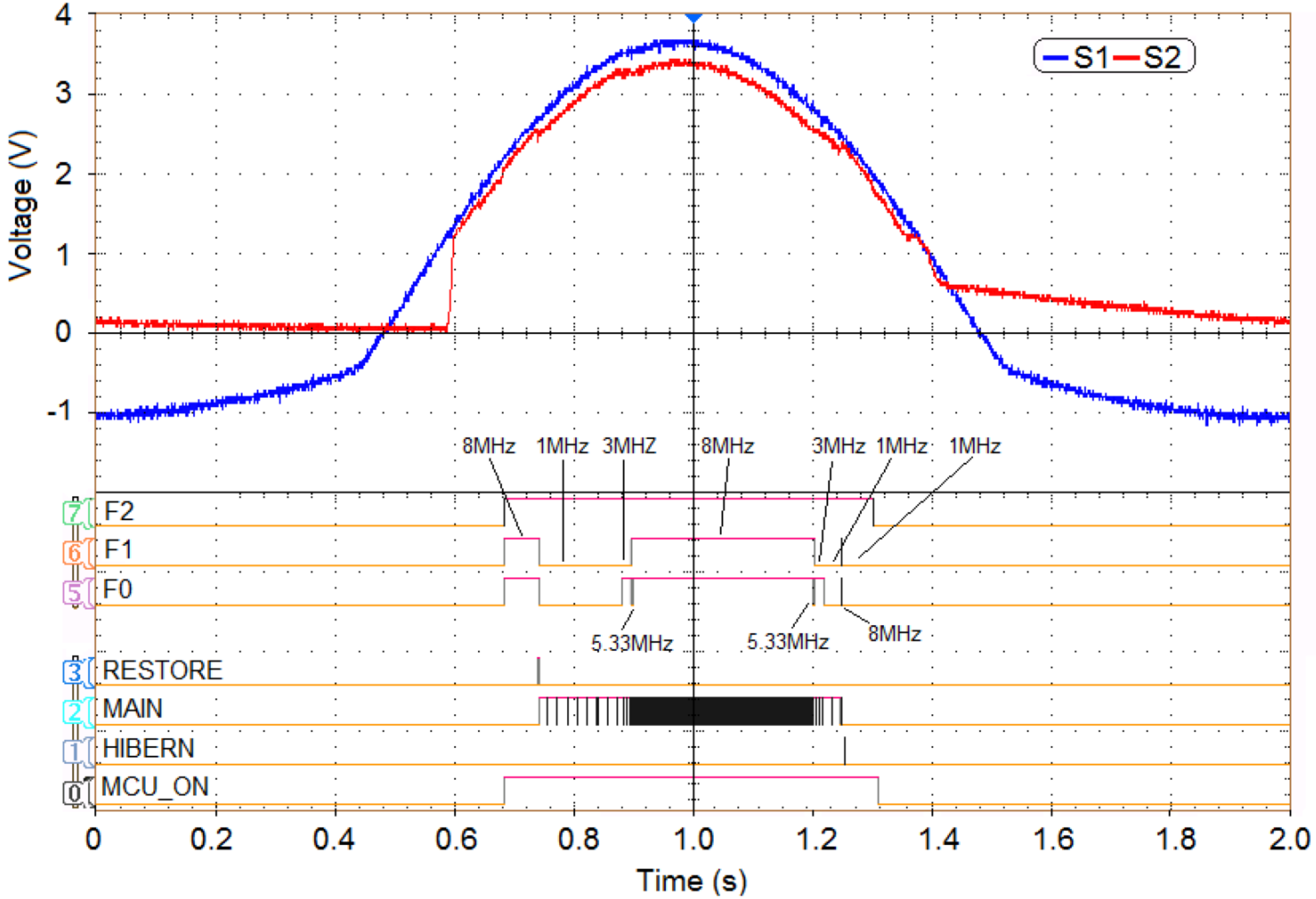
1. Attempts to degrade performance to match source power
2. If it’s unable, the system hibernates as previous





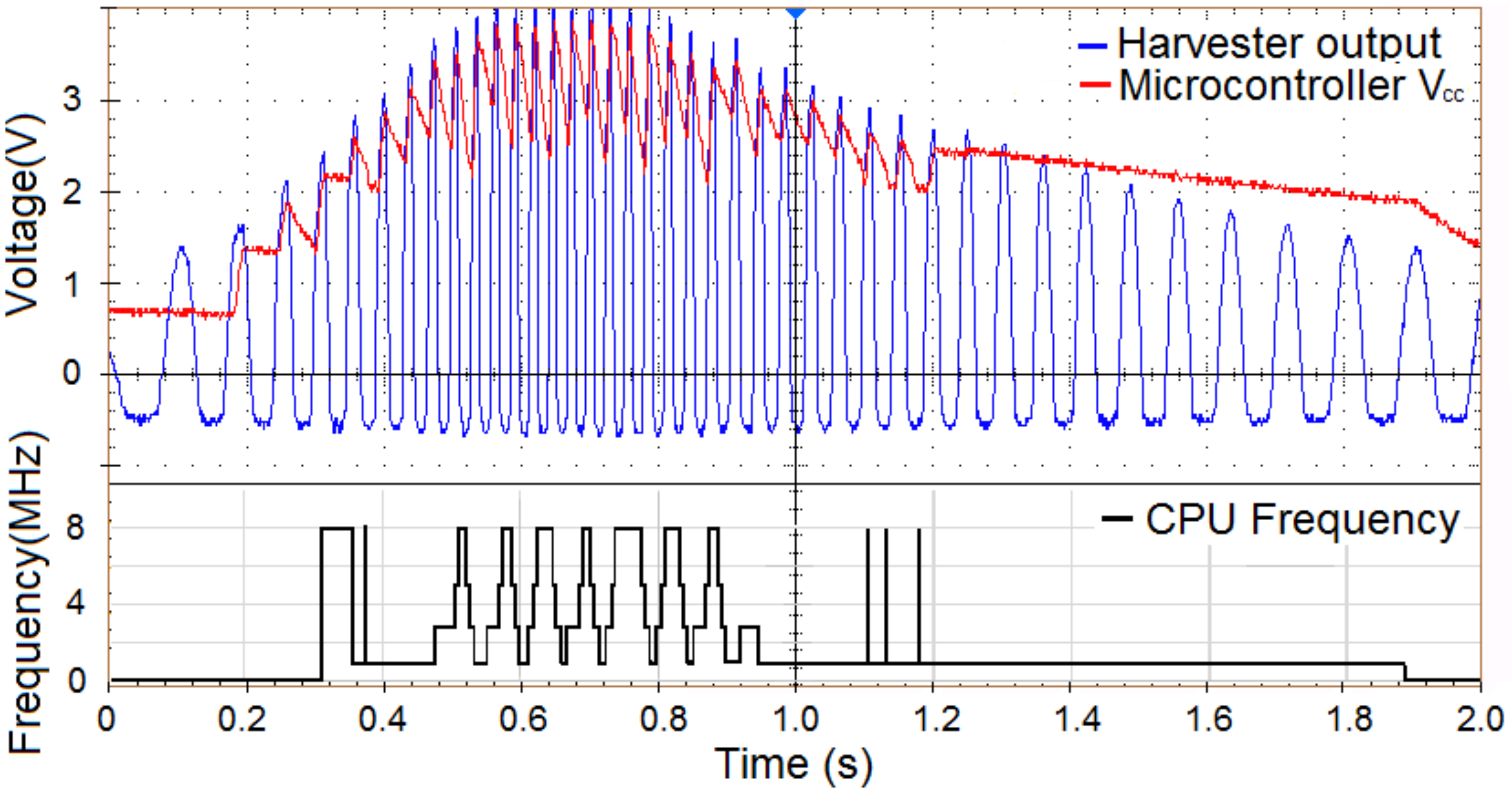
D. Balsamo, A. Das, A.S. Weddell, D. Brunelli, B.M. Al-Hashimi, G.V. Merrett, L. Benini, (2016) Graceful Performance Modulation for Power-Neutral Transient Computing Systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*

Results (Response to Sine Wave)



D. Balsamo, A. Das, A.S. Weddell, D. Brunelli, B.M. Al-Hashimi, G.V. Merrett, L. Benini, (2016) Graceful Performance Modulation for Power-Neutral Transient Computing Systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*

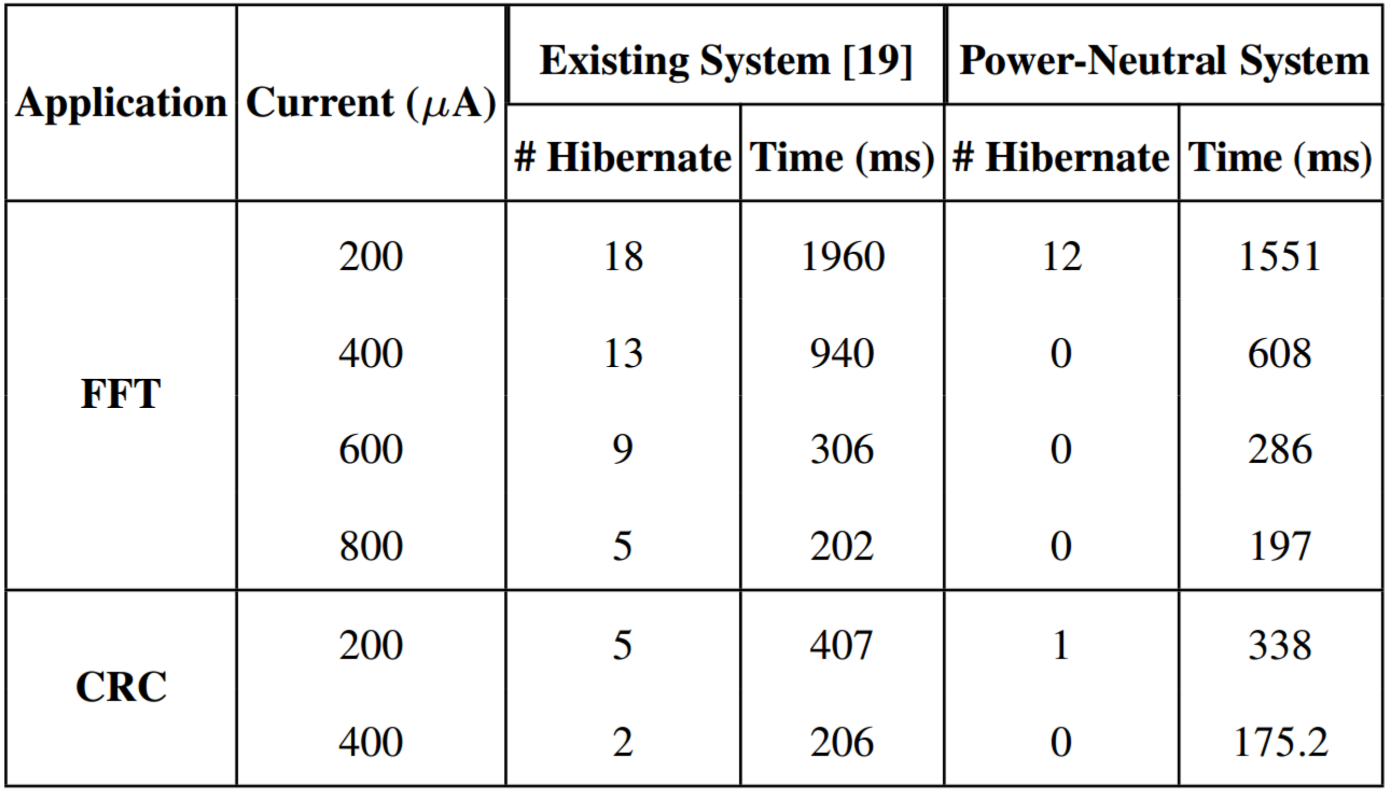
Results (Response to Wind Turbine)



D. Balsamo, A. Das, A.S. Weddell, D. Brunelli, B.M. Al-Hashimi, G.V. Merrett, L. Benini, (2016) Graceful Performance Modulation for Power-Neutral Transient Computing Systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*

Power Neutral Benefits

- Total execution from different constant current sources



Application	Current (μA)	Existing System [19]		Power-Neutral System	
		# Hibernate	Time (ms)	# Hibernate	Time (ms)
FFT	200	18	1960	12	1551
	400	13	940	0	608
	600	9	306	0	286
	800	5	202	0	197
CRC	200	5	407	1	338
	400	2	206	0	175.2

D. Balsamo, A. Das, A.S. Weddell, D. Brunelli, B.M. Al-Hashimi, G.V. Merrett, L. Benini, (2016) Graceful Performance Modulation for Power-Neutral Transient Computing Systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*

Conclusions

a paradigm shift?

Energy-Harvesting Systems

Operating from harvested energy

- *A lot of potential, particularly in specific application domains*
- *A lot of challenges still to overcome*
- ***Inherently different to battery or mains powered systems***

Energy-Neutral Computing

Buffering energy for a battery-like supply

- *Great, for lots of scenarios*
- *Increases size, volume, cost etc*
- ***Over-engineered [evolution not revolution]?***

Transient Computing

Computation when power is available

- *Re-thinking the way we design EH systems*
- ***But need to rethink the whole system***
  - *Peripherals*
  - *Sensors*
  - *Radio communications*
  - *The application!*

Power-Neutral Computing

Adaptive computation when power is available

- ***Complementary to transient computing***
- *Need to get better power-proportionality*

Thank you!

Any Questions?

Dr Geoff Merrett
Associate Professor

Electronic and Software Systems
Tel: +44 (0)23 8059 2775
Email: gvm@ecs.soton.ac.uk www.geoffmerrett.co.uk
Highfield Campus, Southampton, SO17 1BJ UK