Worst case energy consumption of programs at ISA and LLVM IR level

K. Georgiou[1], **J. Morse**[1], K. Eder[1],

[1]University of Bristol

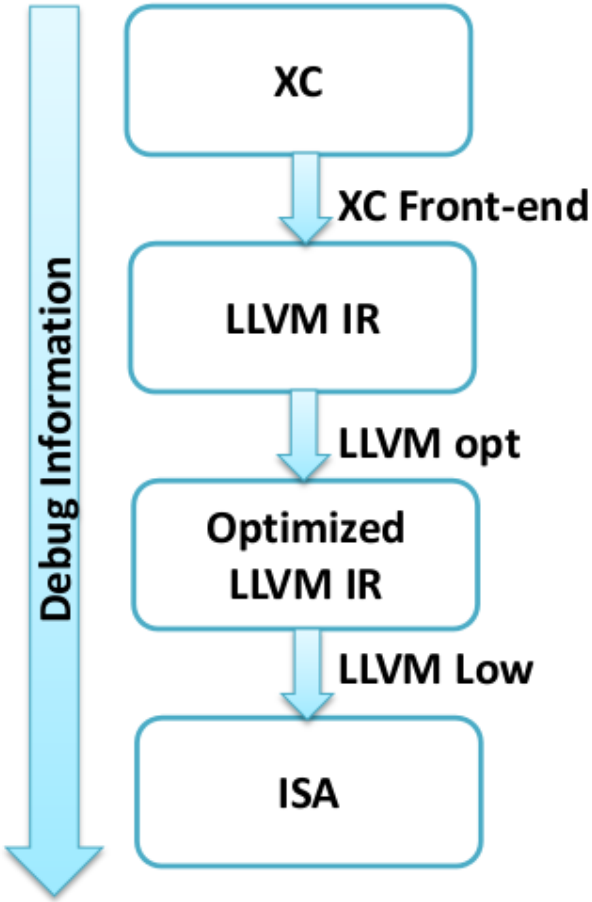
7th EACO Workshop

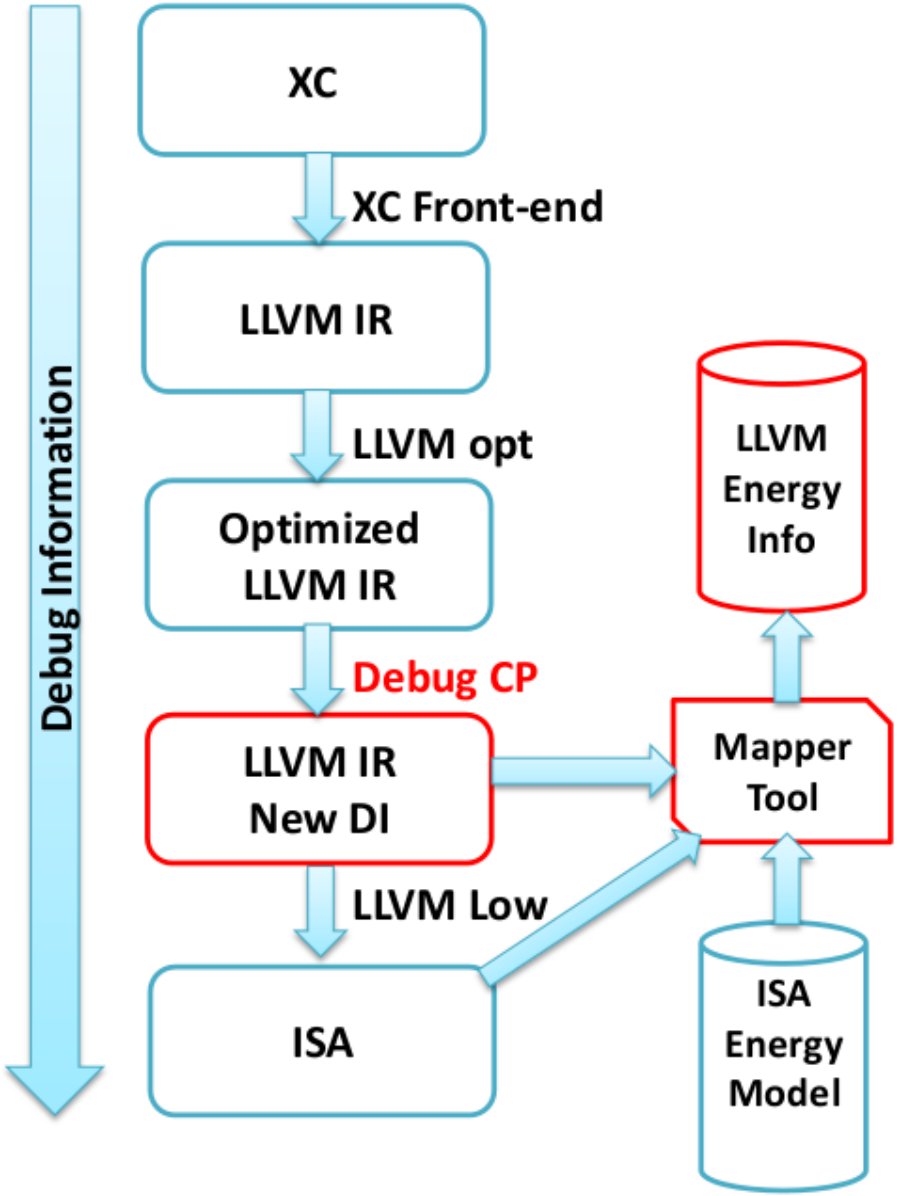
September 11, 2014

funded by

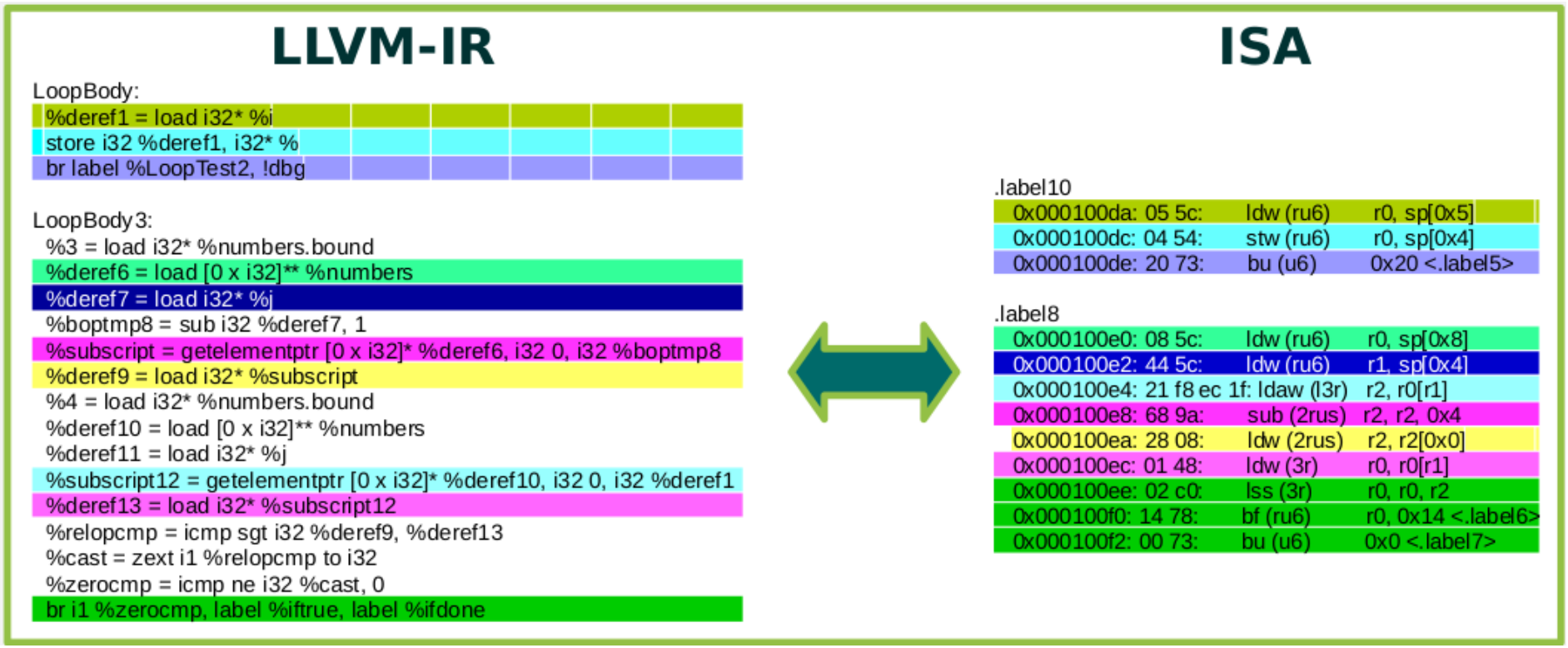
Overview

Mapping Technique





LLVM-IR/ ISA Mapping Example



Approach 1 - Building Cost Functions

Original program

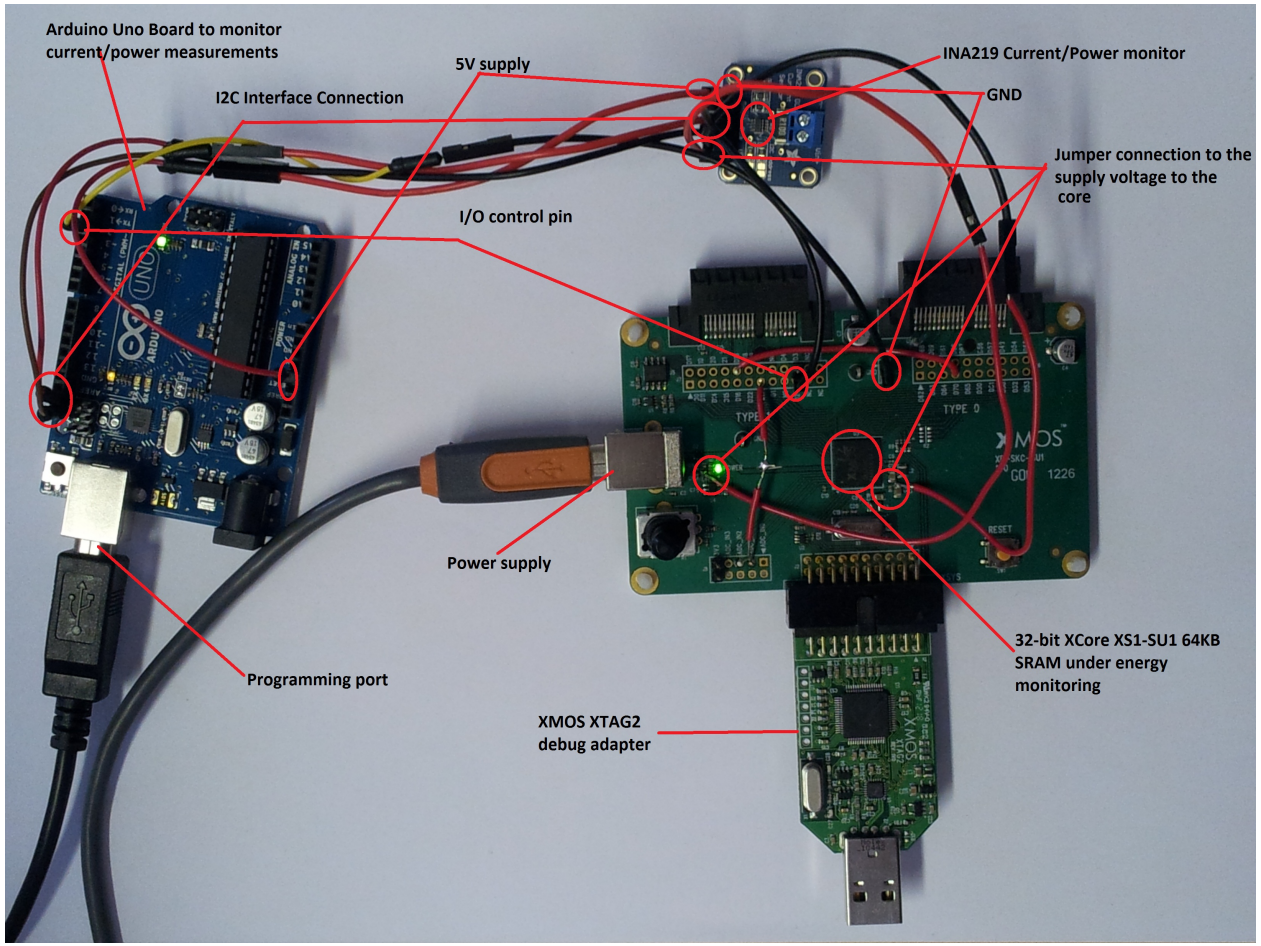
```
int fact(int i) {
  if (i<=0)^a
    return 1^b;
  return (i *^d fact(i-1))^c
      ;
```

Extracted cost relations

$$C_{fact}(i) = C_a + C_b \qquad \text{if } i <= 0$$
$$C_{fact}(i) = C_a + C_c(i) \qquad \text{if } i > 0$$
$$C_c(i) = C_d + C_{fact}(i-1)$$

- Substitute C_a, C_b, C_d with actual energy required to execute low level instructions.
- Solve relations using off the shelf solvers to obtain *closed form* solution.
- Result: $C_{fact}(i) = 4563 + 7878i \ pJ$.

Energy Measuring Set Up



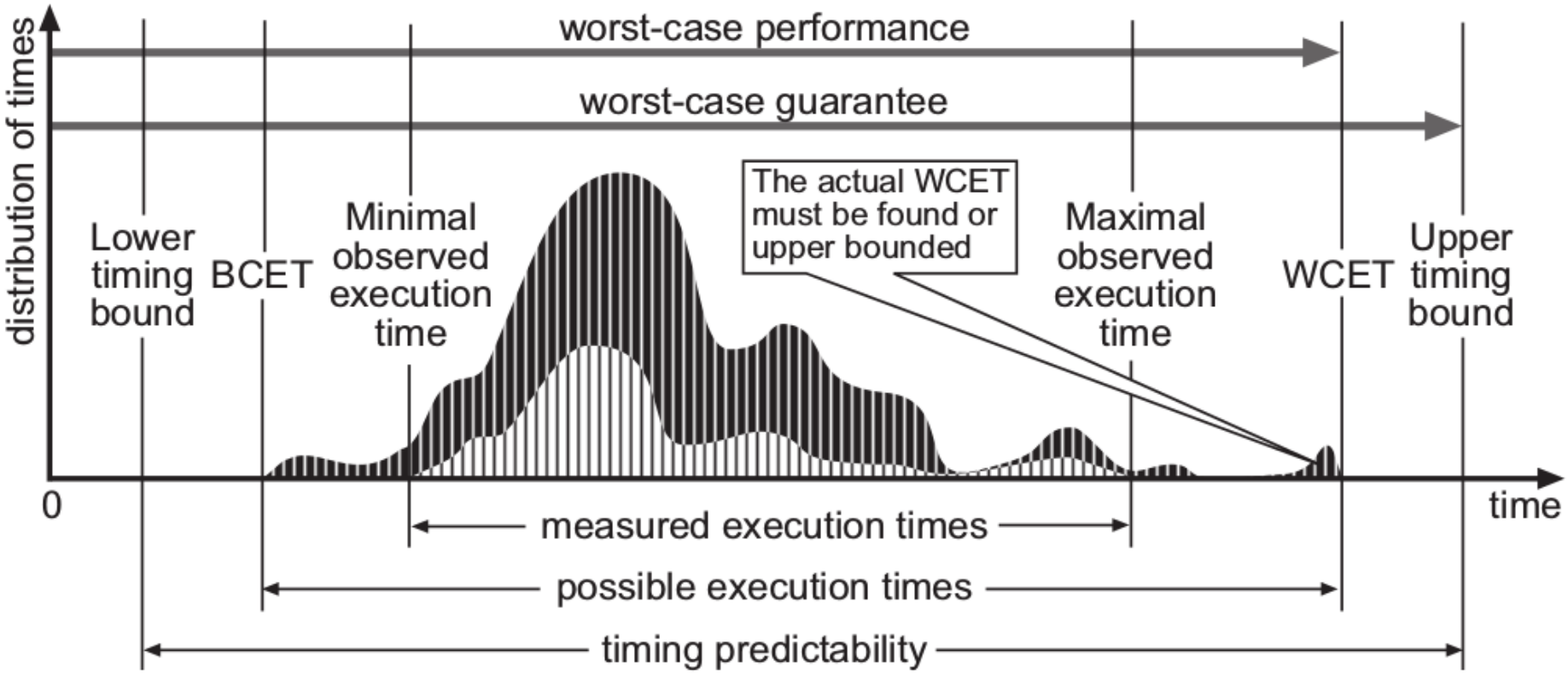
Results

Benchmark*	Formulae		Final error (%)	
	ARM (nJ)	XMOS (nJ)	ARM	XMOS
base64	$158 + 94 \cdot \left\lfloor \frac{P-1}{3} \right\rfloor$	$1270 + 734 \cdot \left\lfloor \frac{P-1}{3} \right\rfloor$	28.0	1.1
mac	$23P + 14$	$133P + 192$	-1.7	10.1
insertion sort	$25P^2 + 11P + 7.1$	$105P^2 + 30P + 75$	11.1	3.0
matrix multiply	$20P^3 + 13P^2 + 97P + 84$	$144P^3 + 200P^2 + 77P + 332$	-3.3	-3.4
jpegdct	54 mJ‡	463 mJ‡	8.5	2.6
Mean relative error			9.9	3.4

Worst Case Execution Time

- How is this related to Energy Consumption?
  - Is time and energy the same?
  - Critical when optimizing between various resources
  - Essential for time critical applications
  - Essential for inferring Energy Consumption of concurrent programs

Bounding WCET



[1] Wilhem et al. The Worst-case Execution-time Problem&Mdash;Overview of Methods and Survey of Tools. In Proceedings of ACM Trans. Embed. Comput. Syst., pages 36:1–36:53, 2008.

ISA Instruction Time Cost

- One ISA instruction needs 4 clock cycles to complete*
- Cycle time

$$T_{clk} = 1/F$$

- Instruction Time

$$I_t = T_{clk} * 4$$

- e.g. 400 MHz $\Rightarrow I_t = 10ns$

***up to four threads**

ISA Instruction Time Cost Exceptions

- Division
- Communication Time is constant on the same core
- Communication Time btwn cores(Steve's Network Modeling)
- Input output on ports time may vary

Implicit Path Enumeration (IPE)

- Very popular technique for WCET calculation
- **It expresses the search of the WCET as an Integer Linear Programming problem where the execution time is to be maximized under some constraints on the execution counts of the basic blocks**
- **The worst case execution path is defined by the set of blocks with their respective execution counts but not the order which they are executed**

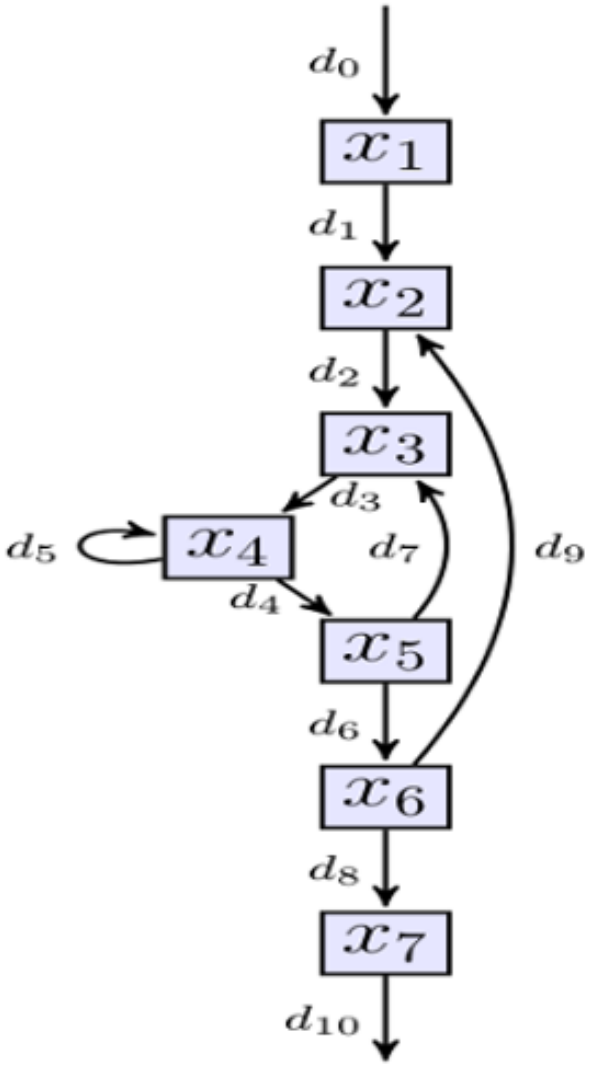
ILP Program Functional Constraints

- Constraints used to denote loop bounds and other path information that depend on the functionality of the program (Data Flow Analysis can minimize user input)
- Minimum requirement from programmer to set the loop bounds
- More functional constraints from the user can help to get tighter bounds

JpegDCT Code

```
void jpegdct(short d[], short r[])
{
     long int t[12];
     int v=0;
     short i, j, k, m, n, p, ic, ik;
     for (ik=2; ik; ik--) {
       for (i = 8; i; i--, v+=8) {
            for (j = 3; j>=0; j--) {
                 // some code
             }
           // some code
        }
     }
}
```

ILP Program F. Constraints Loop Bounds Example



$$1x1 <= x2 \quad (1)$$

$$x2 <= 2x1 \quad (2)$$

$$1x2 <= x3 \quad (3)$$

$$x3 <= 8x2 \quad (4)$$

$$1x3 <= x4 \quad (5)$$

$$x4 <= 4x3 \quad (6)$$

ILP Solving Example

$max : b_1 * x1 + b_2 * x2 + b_3 * x3 + b_4 * x4 + b_5 * x5 + b_6 * x_6 + b_7 * x_7$

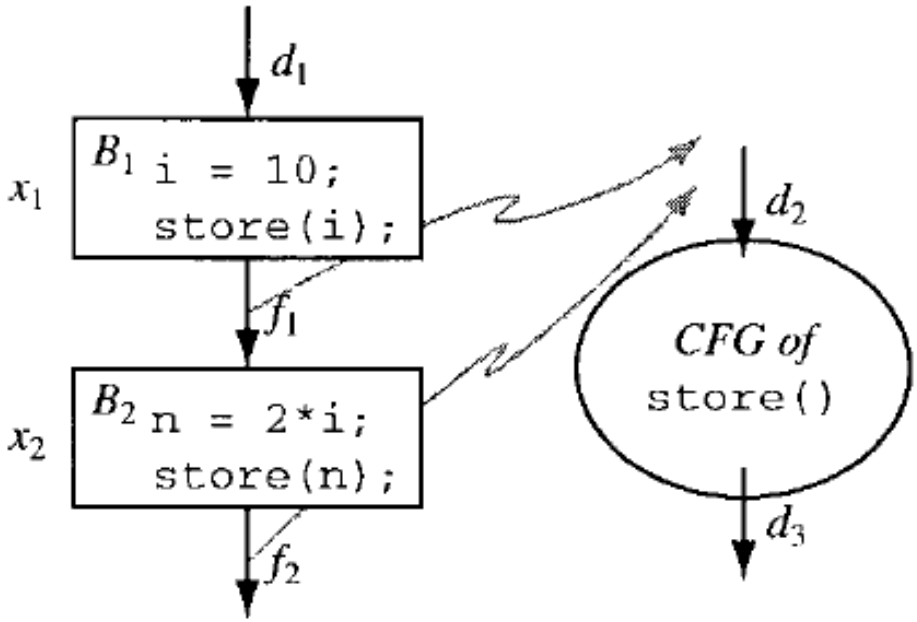
$$d_0 = 1$$
$$x_1 = d_1 = d_0$$
$$x_2 = d_1 + d_9 = d_2$$
$$x_3 = d_2 + d_7 = d_3$$
$$x_4 = d_3 + d_5 = d_4 + d_5$$
$$x_5 = d_4 = d_6 + d_7$$
$$x_6 = d_6 = d_8 + d_9$$
$$x_7 = d_8 = d_{10}$$
$$d_{10} = 1$$

$$1x1 <= x2$$
$$x2 <= 2x1$$
$$1x2 <= x3$$
$$x3 <= 8x2$$
$$1x3 <= x4$$
$$x4 <= 4x3$$

- Solve this by lp_solver: standard linux pri-installed package
- Complexity: NP complete, although most of the cases it collapses to LP which can be solved in polynomial time

ILP Program S. Constraints Function Call Example

f-edges treated similar to d-edges

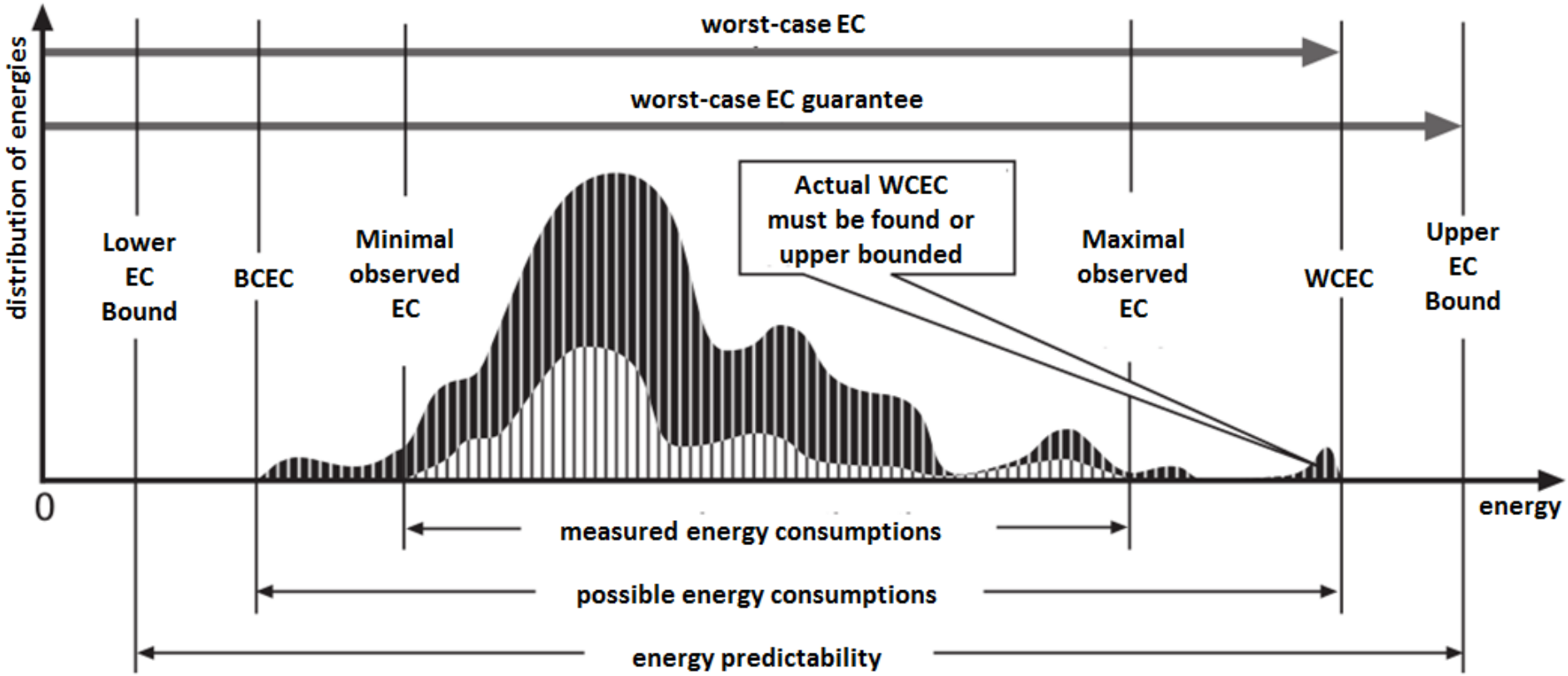


$$x_1 = d_1 = f_1$$

$$x_2 = f_1 = f_2$$

$$d_2 = f_1 + f_2$$

Worst Case Energy Consumption(WCEC)



[1] Modified version of: Wilhem et al. The Worst-case Execution-time Problem&Mdash;Overview of Methods and Survey of Tools. In Proceedings of ACM Trans. Embed. Comput. Syst., pages 36:1–36:53, 2008.

Calculating WCEC bounds

- Using the same approach used for time to formulate and solve the problem by using ILP
- Replace the timing cost B_i of the basic blocks on the CFG with their energy cost
- Solve the retrieved equation with its' constraints by maximizing and minimizing it to get the upper and lower bounds of the energy consumption
- Currently retrieving CFGs and the ILP formulation and constraints automatically in both ISA and LLVM IR

Experimental Results

Benchmark	HW VS ISA
Mac	-2.5%
Matrix Mul	-1.1%
Levenshtein	-0.7%
Jpegdct	-2.3%
Base64	-2.2%
Radix4Division	45%

Conclusion and Future Work

- WCET techniques can be used for WCEC
- If you need predictability make it predictable
  - predictable architecture
  - predictable software
- Compine this with abstract interpretation
- Try this with an arm Cortex M3
- Extend this to multi-threaded case

Swallow Project

WCEC Models

- The program path length can scale with input data (and can be analysed by WCET)
- However, energy consumption of individual instructions can scale with input data too
- In Steve's XCore energy model, `lmul` can range from consuming energy at 131mW to 222mW, depending on the value of it's operands
- Assuming the worst-case consumption for each instruction is safe, but not a particularly tight approximation.

Data Path Analysis

```
int fir(int xn, int coeffs[], int state[], int ELEMS)
{
  int ynl = 0, ynh = (1<<23);
  for (j = ELEMS-1; i != 0; j--) {
    state[j] = state[j-1];
    {ynh,ynl} =
        maccs(coeffs[j], state[j], ynh, ynl);
  }
}
```

```
sub         r6, r5, 0x1
ldw         r7, r2[r6]
stw         r7, r2[r5]
ldw         r5, r1[r5]
maccs       r0, r3, r5, r7
add         r5, r7, 0x0
bt          r6, -0xF <label>
```

Work Plan

- Verify that power consumption scales with the frequency of instructions operating on the data path
- Data flow or taint analysis to identify that data path
- Integration into ENTRA project toolchain
- Evaluation

Thank you!

kyriakos.georgiou@bristol.ac.uk

jeremy.morse@bristol.ac.uk

kerstin.eder@bristol.ac.uk