

## High-level Synthesis from the Synchronous Language Esterel

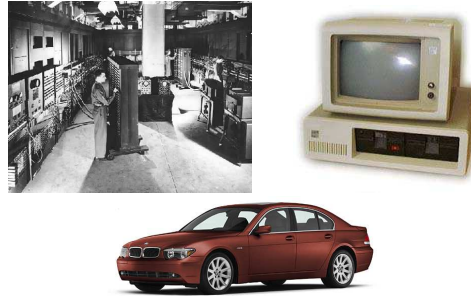
### 2004 MDC Conference

Stephen A. Edwards

Columbia University

High-level Synthesis from the Synchronous Language Esterel - p. 12

## Spot the Computer

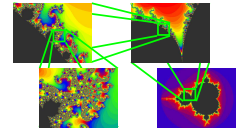


High-level Synthesis from the Synchronous Language Esterel - p. 22

## Technical Challenges



Real-time



Complexity



Concurrency



Legacy Languages

Photo by Thomas Banopica

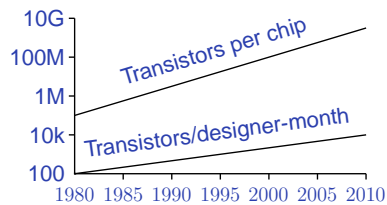
High-level Synthesis from the Synchronous Language Esterel - p. 32

## Motivation: Rising Design Cost

1981: 100 designer-months for leading-edge chip  
10k transistors, 100 transistors/month

2002: 30 000 designer-months  
150M transistors, 5000 transistors/month

Design cost increased from \$1M to \$300M



High-level Synthesis from the Synchronous Language Esterel - p. 42

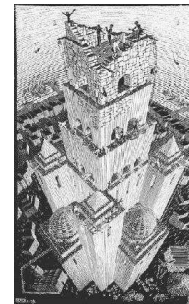
## Domain-Specific Languages

Little languages that fit the problem

More succinct description that are

1. Quicker to create
2. Easier to get right

More opportunities for optimization and analysis  
General-purpose languages hindered by undecidability  
Domain-specific languages much simpler



High-level Synthesis from the Synchronous Language Esterel - p. 52

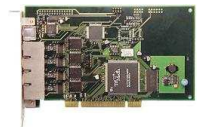
## Languages for Device Drivers

Device drivers are those pieces of software that you absolutely need that never seem to work

Big security/reliability hole: run in Kernel mode

Responsible for 80% of all Windows crashes  
Tedious, difficult-to-write

Ever more important as customized hardware proliferates



High-level Synthesis from the Synchronous Language Esterel - p. 62

## Ongoing Work

Develop language for network card drivers under Linux (Chris Conway)

Sharing drivers between Linux and FreeBSD (Tom Heydt-Benjamin)

Ultimate vision: compiler takes two programs: device spec. and OS spec. and synthesizes appropriate driver.

OS vendor makes sure OS spec. is correct;  
Hardware designer makes sure hardware spec. is correct.

## NE2000 Ethernet driver (fragment)

```
ioports ne2000 {
  bits cr {
    bit stop, sta, transmit;
    enum:3 { 001=remRead, 010=remWrite,
            011=sendPacket, 1**=DMAdone }
  }
  enum:2 { 00=page0, 01=page1, 10=page2 }
}
paged p {
  page0 { cr.page0; } {
    twobyte clda;
    byte bnry;
    bits tsr {
      bit ptx,1,col,abt,crs,0,cdh,owc;
    }
  }
  page1 { cr.page1; } {
    byte:6 par;
    byte curr;
    byte:8 mar;
  }
}
```

## The Esterel Real-Time Language

Synchronous language developed by Gérard Berry in France

Basic idea: use global clock for synchronization in software like that in synchronous digital hardware.

Challenge: How to combine concurrency, synchronization, and instantaneous communication



## An Overview of Esterel

Synchronous model of time: implicit global clock

Communication through wire-like signals

Two flavors of statement:

| Combinational               | Sequential                  |
|-----------------------------|-----------------------------|
| <i>Execute in one cycle</i> | <i>Take multiple cycles</i> |
| emit                        | pause                       |
| present / if                | await                       |
| loop                        | sustain                     |

High-level Synthesis from the Synchronous Language Esterel -- p. 102

## An Example

```
emit B;
present C then
  emit D end;
```

Force signal present in this cycle  
Make D present if C is

High-level Synthesis from the Synchronous Language Esterel -- p. 112

## An Example

```
await A;
emit B;
present C then
  emit D end;
pause
```

Wait for next cycle where A is present  
Wait for next cycle

High-level Synthesis from the Synchronous Language Esterel -- p. 122

## An Example

```
loop
  await A;
  emit B;
  present C then
    emit D end;
  pause
end
```

Infinite Loop

High-level Synthesis from the Synchronous Language Esterel -- p. 132

## An Example

```
loop
  await A;
  emit B;
  present C then
    emit D end;
  pause
end
||
loop
  present B then
    emit C end;
  pause
end
```

Run Concurrently

High-level Synthesis from the Synchronous Language Esterel -- p. 142

## An Example

```
every R do
  loop
    await A;
    emit B;
    present C then
      emit D end;
    pause
  end
  ||
  loop
    present B then
      emit C end;
    pause
  end
end
```

Restart on R

High-level Synthesis from the Synchronous Language Esterel -- p. 152

## An Example

```
every R do
  loop
    await A;
    emit B;
    present C then
      emit D end;
    pause
  end
  ||
  loop
    present B then
      emit C end;
    pause
  end
end
```

Same-cycle bidirectional communication

## An Example

```
every R do
  loop
    await A;
    emit B;
    present C then
      emit D end;
    pause
  end
  ||
  loop
    present B then
      emit C end;
    pause
  end
end
```

Good for hierarchical FSMs  
Bad at manipulating data  
Esterel V7 variant proposed to address this

## Why Consider Esterel for Hardware?

- Semantics more abstract than RTL  
More succinct: easier to write faster
- High-level semantics enable optimizations  
State assignment a hierarchical problem
- Semantics enable efficient simulation  
No event queue  
Closer to an imperative program
- Esterel's semantics are deterministic  
Simulation-synthesis mismatches eliminated

## Applications of Esterel

Systems with complex (non-pipelined) control-behavior:

- DMA controllers
- Cache controllers
- Communication protocols

(Not processors)

High-level Synthesis from the Synchronous Language Esterel -- p. 192

## Verilog More Verbose Than Esterel

```

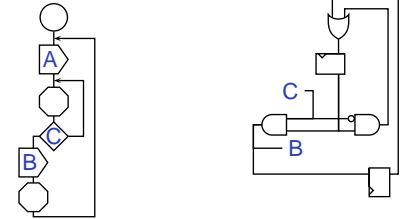
loop
  await
  case [icu_miss and
       not cacheable] do
    emit [normal_ack or error_ack]
  end
  case [icu_miss and
       cacheable] do
    abort
    await 4 normal_ack;
    when error_ack
  end
  case [pcsu_powerdown and
       not jmp_e and
       not valid_diag_window] do
    emit [pcsu_powerdown and
         not jmp_e]
  end
end
pause
end
    
```

High-level Synthesis from the Synchronous Language Esterel -- p. 202

## Basic Circuit Generation

```

loop
  emit A; await C;
  emit B; pause
end
    
```



High-level Synthesis from the Synchronous Language Esterel -- p. 212

## Generating Fast Circuits

Esterel's semantics match hardware. Translation is straightforward.

Nice feature: state space is well-defined and hierarchical (e.g., due to abort and concurrency).

Enables a hierarchical state assignment/synthesis procedure.

High-level Synthesis from the Synchronous Language Esterel -- p. 222

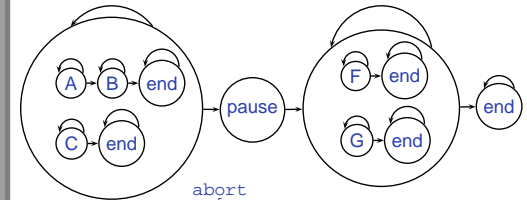
## Hierarchical States

```

abort
[
  await A; await B
  |
  await C
]
when D;
emit E;
pause;
[
  await F
  |
  await G
]
    
```

High-level Synthesis from the Synchronous Language Esterel -- p. 232

## Five Simple FSMs



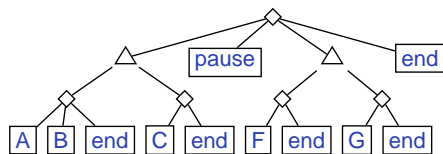
```

abort
[
  await A; await B
  |
  await C
]
when D;
emit E;
pause;
[
  await F
  |
  await G
]
    
```

High-level Synthesis from the Synchronous Language Esterel -- p. 242

## General Problem Statement

States in an Esterel program an arbitrary tree of sequential and parallel state machines.



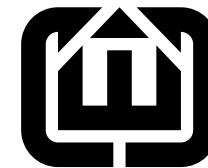
Assign states to local machines to optimize global circuit.

## Results

| Example     | Sis |     |      |        | Xilinx |     |      |      |    |     |    |      |      |     |
|-------------|-----|-----|------|--------|--------|-----|------|------|----|-----|----|------|------|-----|
|             | V5  | CEC | hand | hand   | V5     | CEC | hand | hand |    |     |    |      |      |     |
| Figure 1a   | 23  | 15  | 15   | 6(0)   | 5      | 5   | 4    | 3    | 7  | 4   | 4  | 4.7  | 4.6  | 4.4 |
| daceexample | 41  | 23  | 22   | 7(0)   | 5      | 5   | 5    | 3    | 10 | 5   | 5  | 6.2  | 6.0  | 5.5 |
| jacky1      | 39  | 22  | 20   | 5(0)   | 4      | 4   | 4    | 3    | 6  | 5   | 4  | 5.4  | 6.1  | 5.0 |
| runner      | 218 | 145 | 144  | 30(24) | 20     | 20  | 11   | 10   | 10 | 56  | 36 | 10.6 | 8.4  | 8.1 |
| greycounter | 240 | 173 | 142  | 34(6)  | 18     | 15  | 11   | 13   | 9  | 40  | 34 | 12.4 | 13.4 | 8.9 |
| scheduler   | 519 | 380 |      | 74(52) | 55     |     | 8    | 8    |    | 80  | 66 | 11.3 | 8.9  |     |
| servos      | 407 | 287 |      | 60(16) | 47     |     | 10   | 10   |    | 105 | 66 | 16.7 | 13.4 |     |
| abcd        | 167 | 165 |      | 17(0)  | 13     |     | 7    | 8    |    | 43  | 43 | 12.8 | 12.5 |     |
| tcint       | 508 | 414 |      | 95(14) | 60     |     | 17   | 9    |    | 115 | 81 | 10.8 | 10.9 |     |

20% smaller, run at comparable speeds.  
*Not the final word.*

## The Columbia Esterel Compiler



- Open-Source C++
- Hardware generation
- Software generation

<http://www1.cs.columbia.edu/~sedwards/cec/>