GCC

Marvell Taiwan

Intern Wei-Sheng Chou

MARVELL®

Outline

Chapter1 - Compiler

- Complication … P3
- GCC compiler … P12

Chapter2 - GCC Source Code

- Configuration & Building … P17
- ~~Cross Compile GCC compiler … P21~~

Chapter3 - Gray box probing of GCC

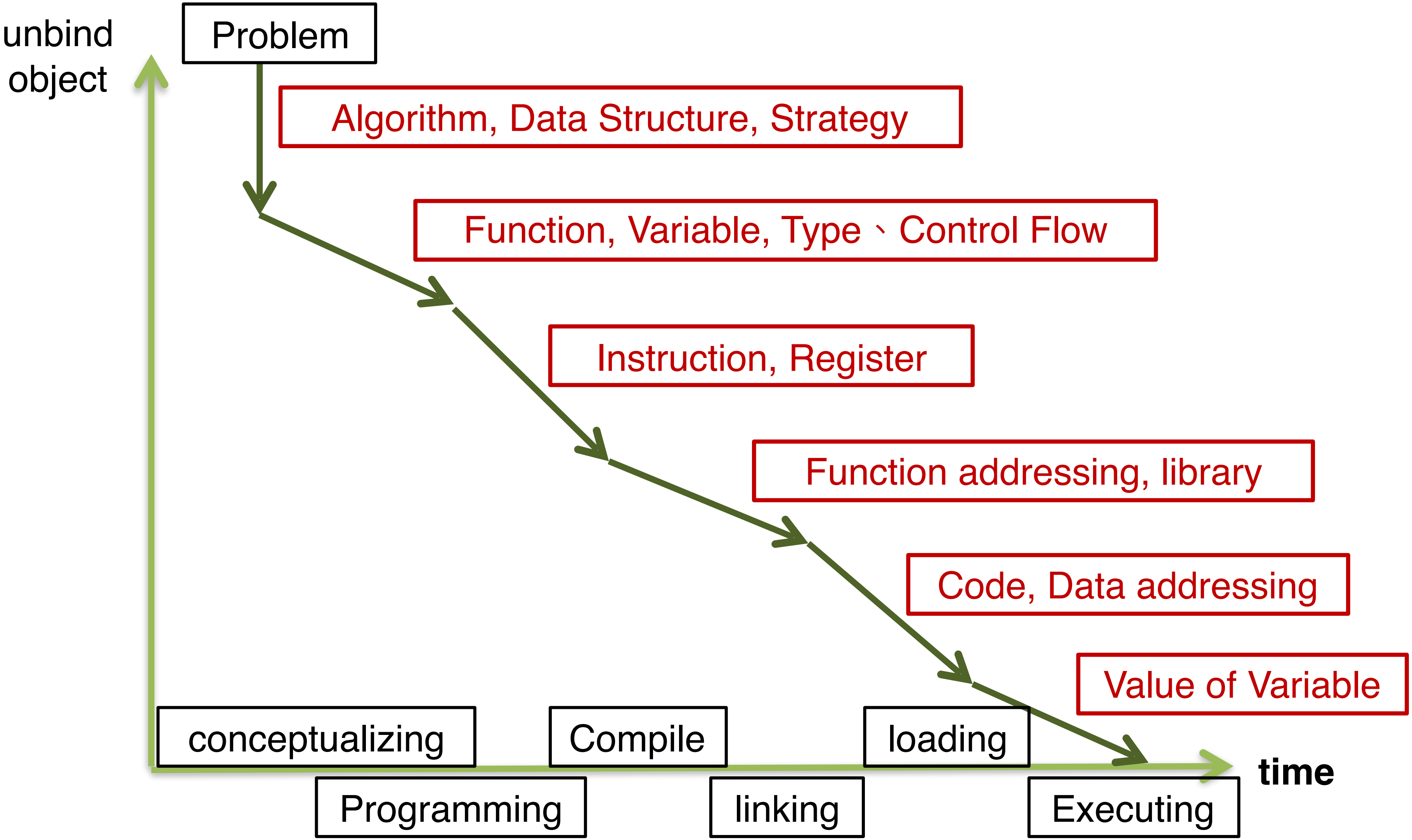
- Gray box probing of GCC … P29
- Passes … P33
- Gimple Optimization … P46

Reference … P57

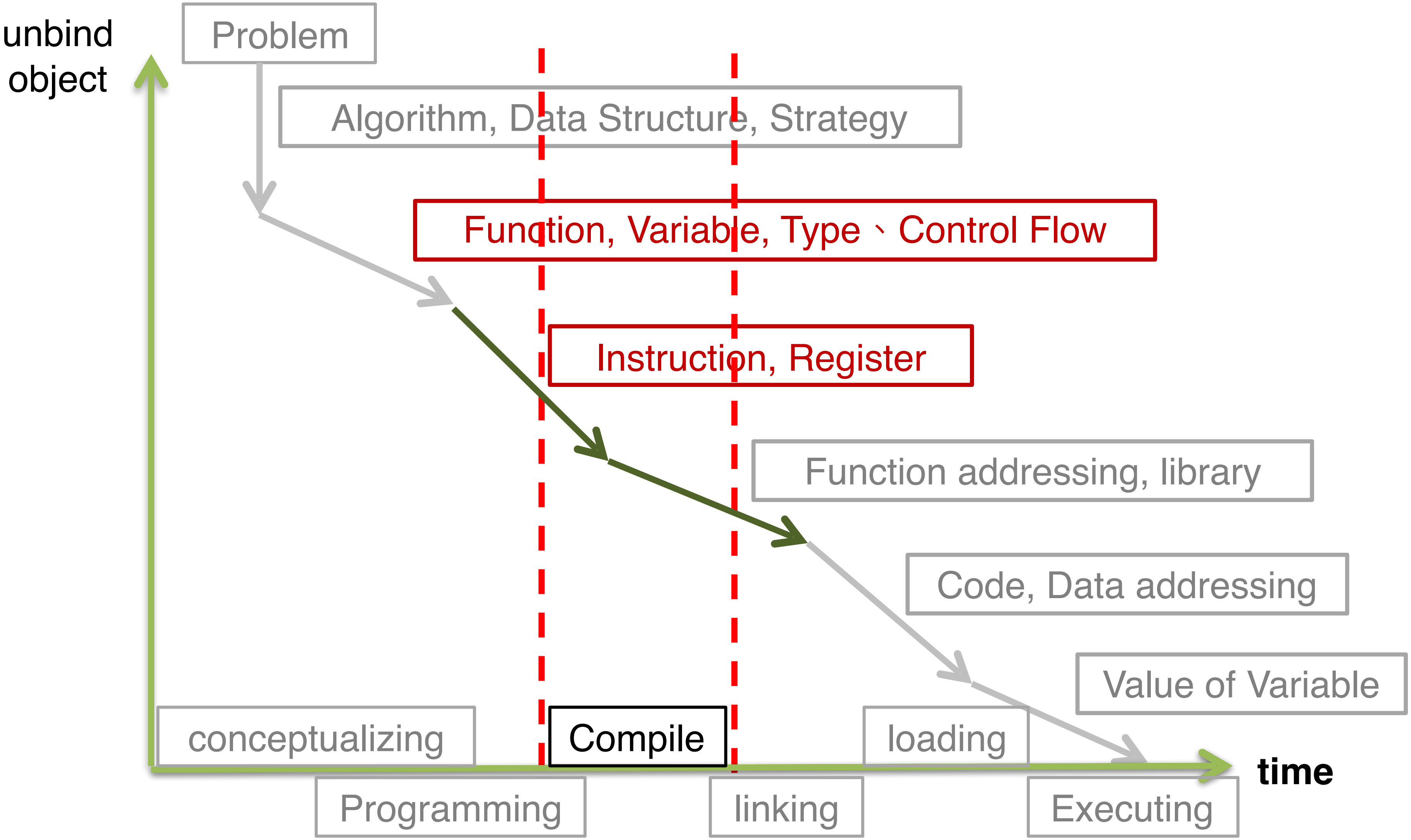
Chapter1

Complication

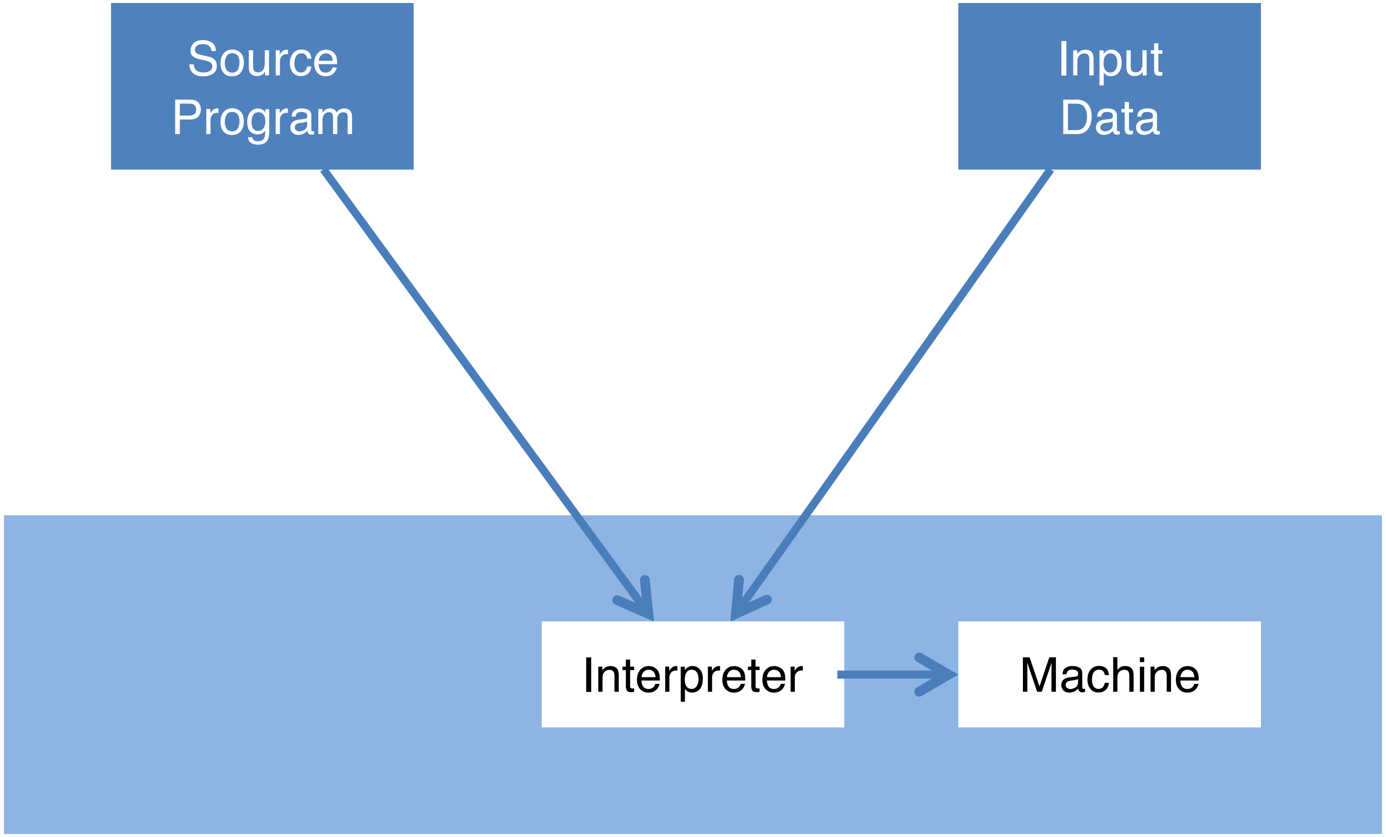
Binding


unbind object
Problem
Algorithm, Data Structure, Strategy
Function, Variable, Type、Control Flow
Instruction, Register
Function addressing, library
Code, Data addressing
Value of Variable
conceptualizing
Compile
loading
time
Programming
linking
Executing

Binding

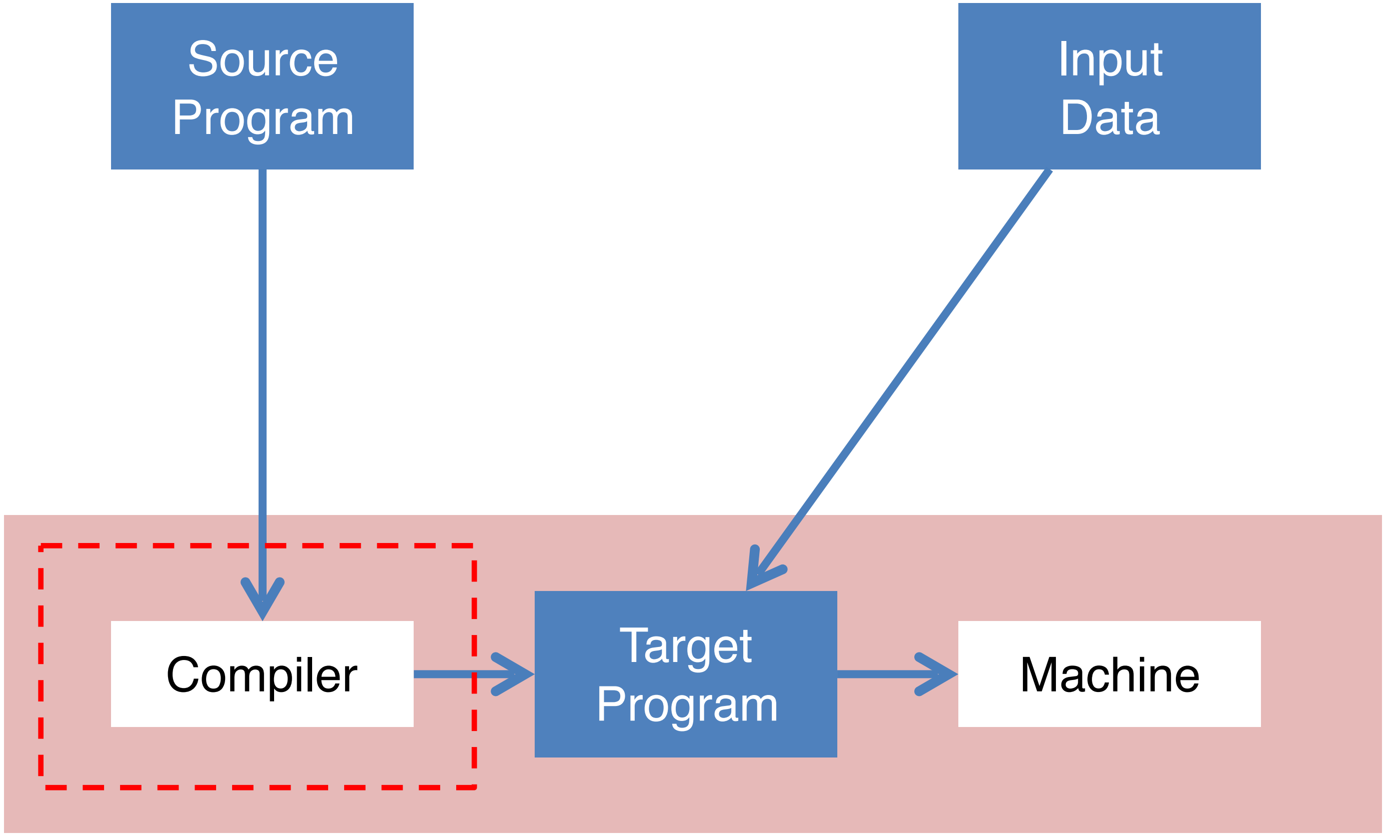

unbind object
Problem
Algorithm, Data Structure, Strategy
Function, Variable, Type、Control Flow
Instruction, Register
Function addressing, library
Code, Data addressing
Value of Variable
conceptualizing
Programming
Compile
linking
loading
Executing
time

Interpreter VS Compiler



Develop
Flexibility

Interpreter VS Compiler

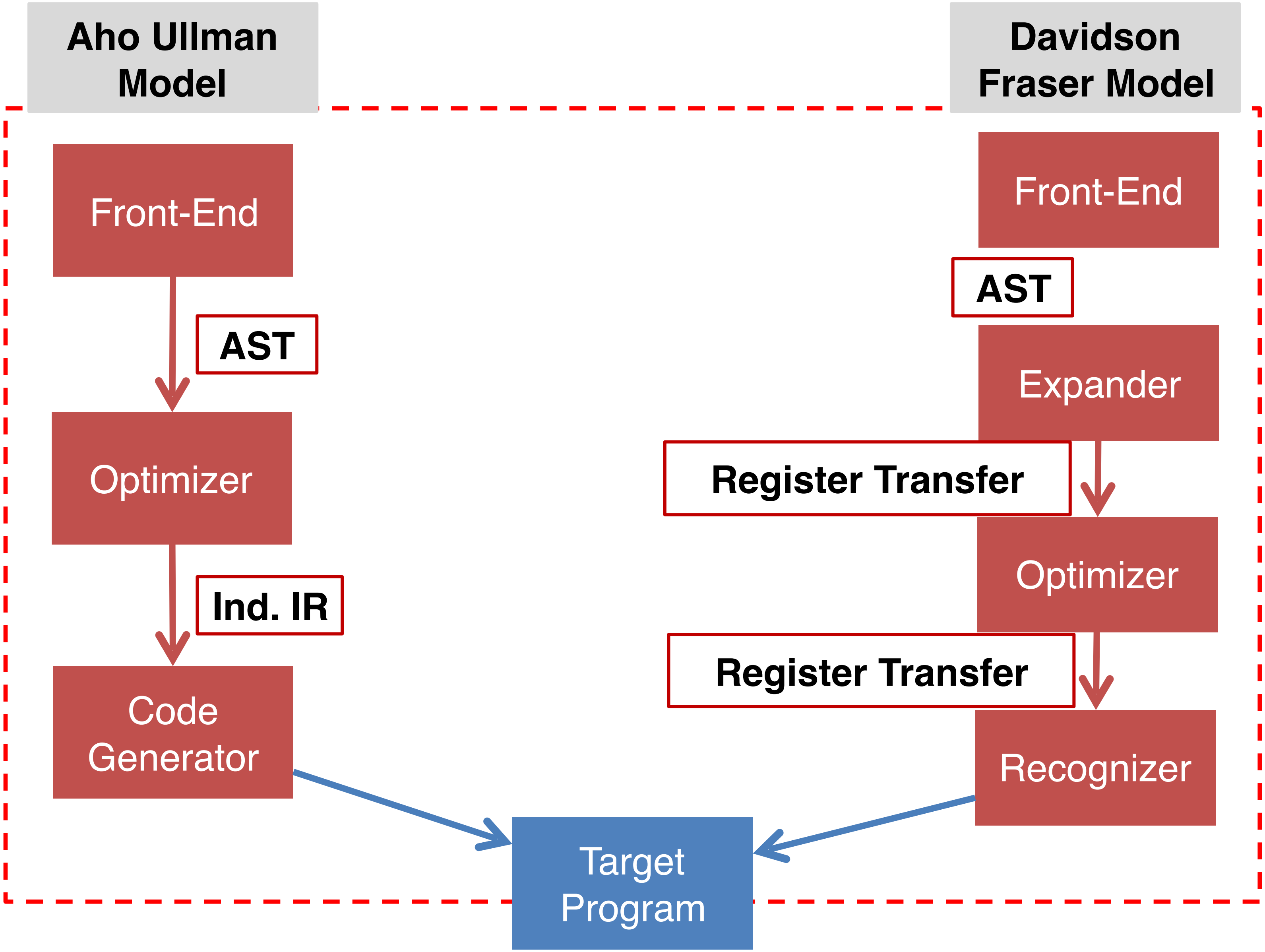


Executing
Speed

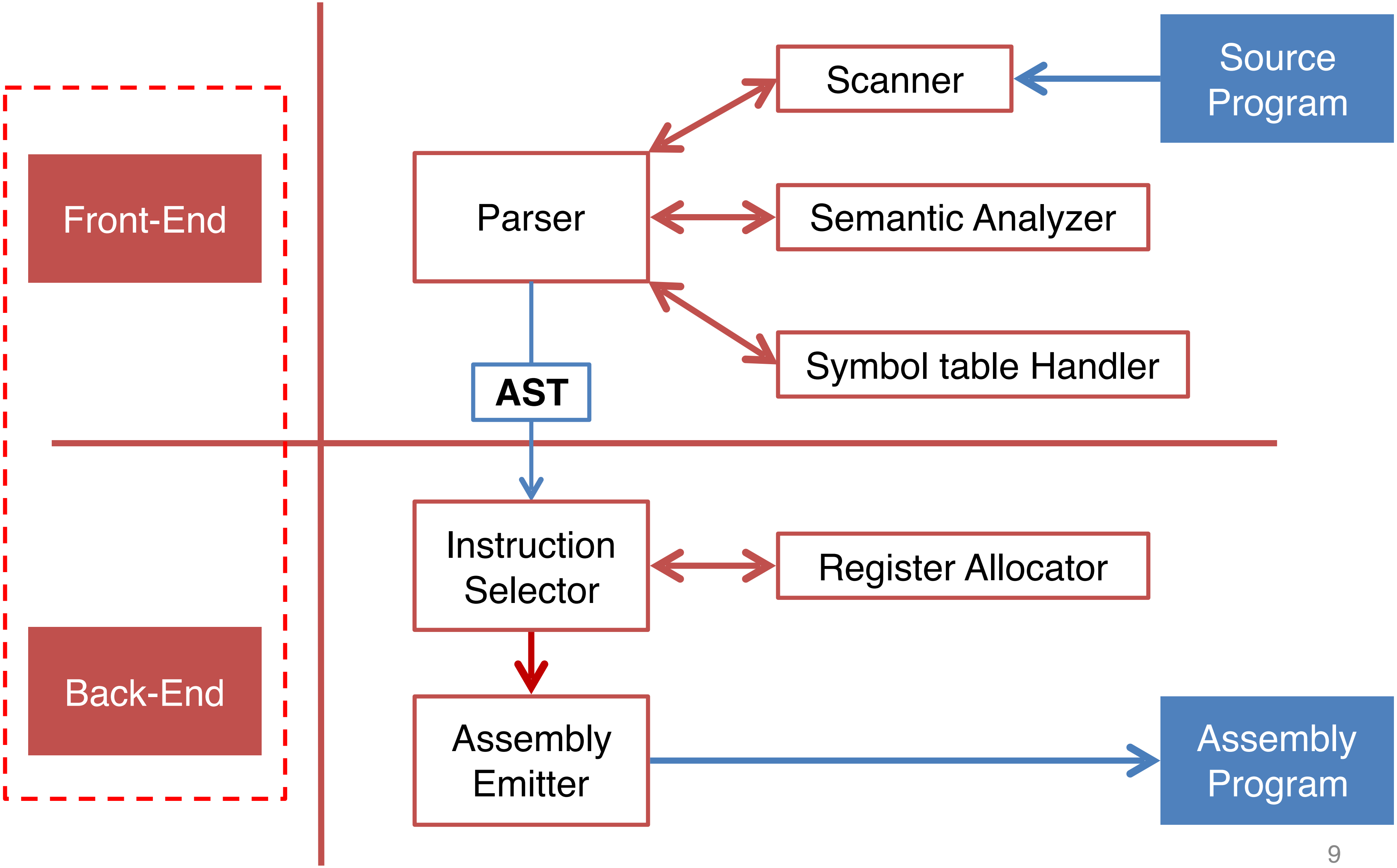
Machine & Optimizer Independent

Model

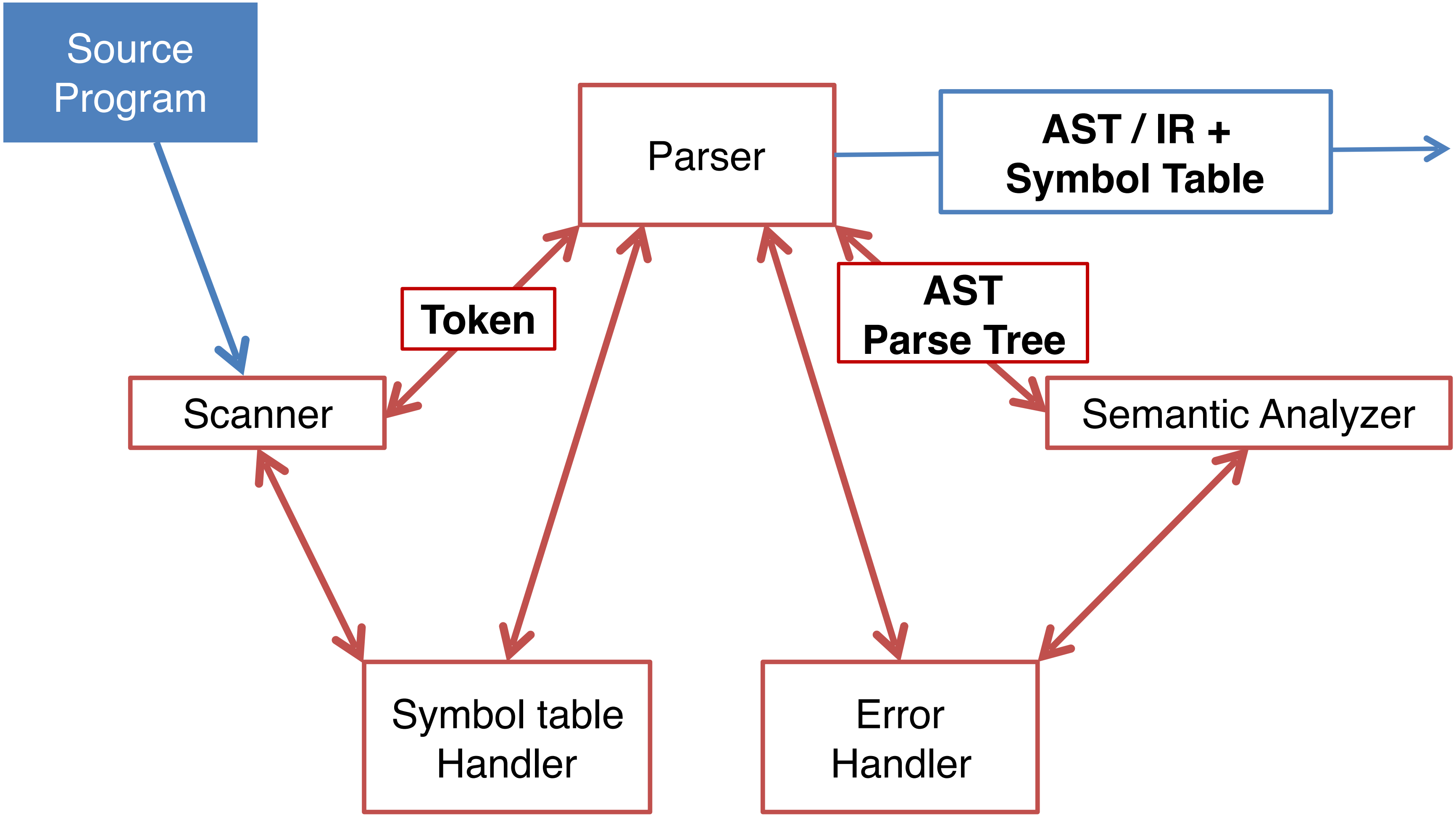
Machine & Optimizer Dependent



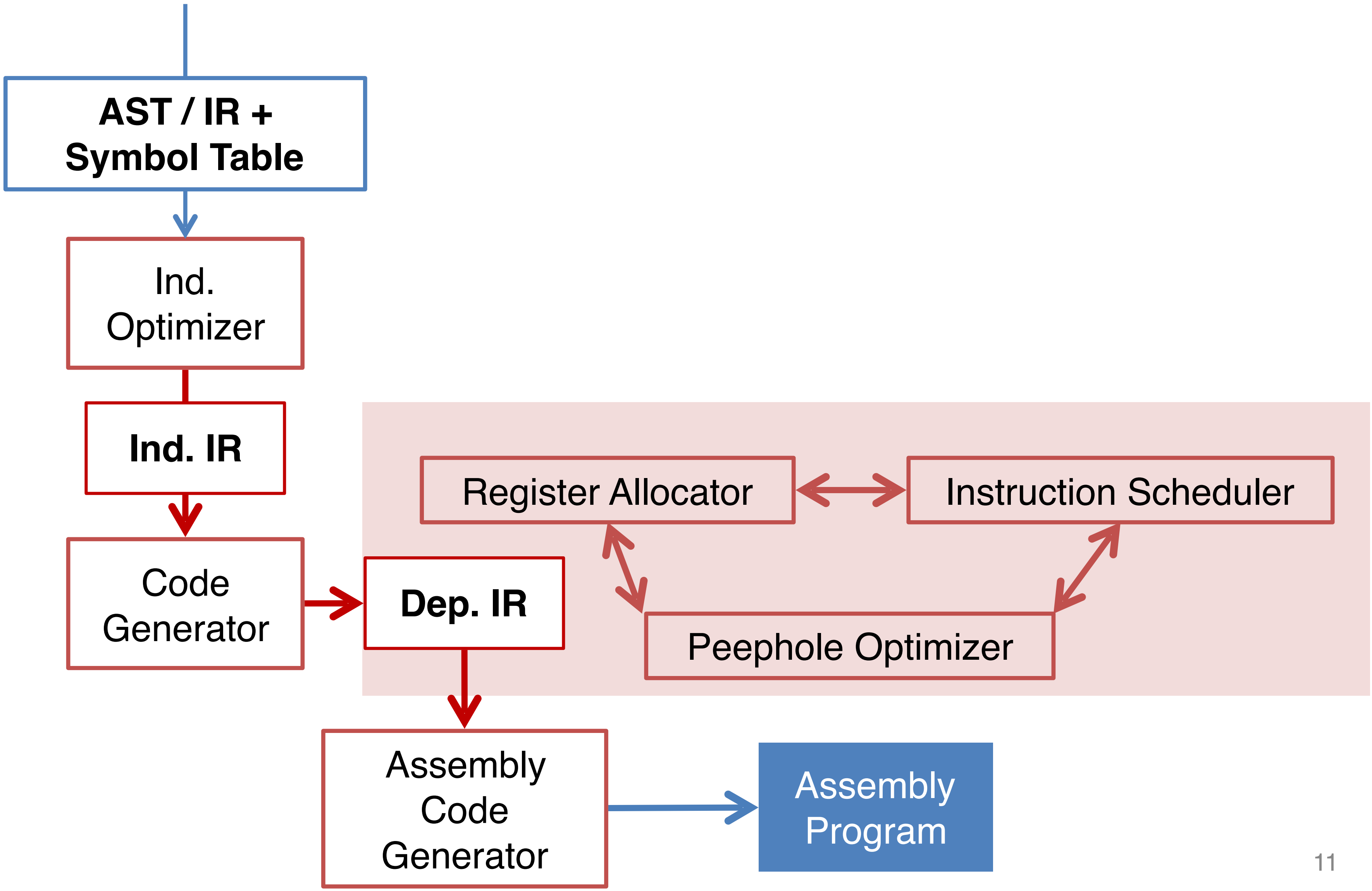
Structure of Compiler



Typical Front-End



Typical Back-End



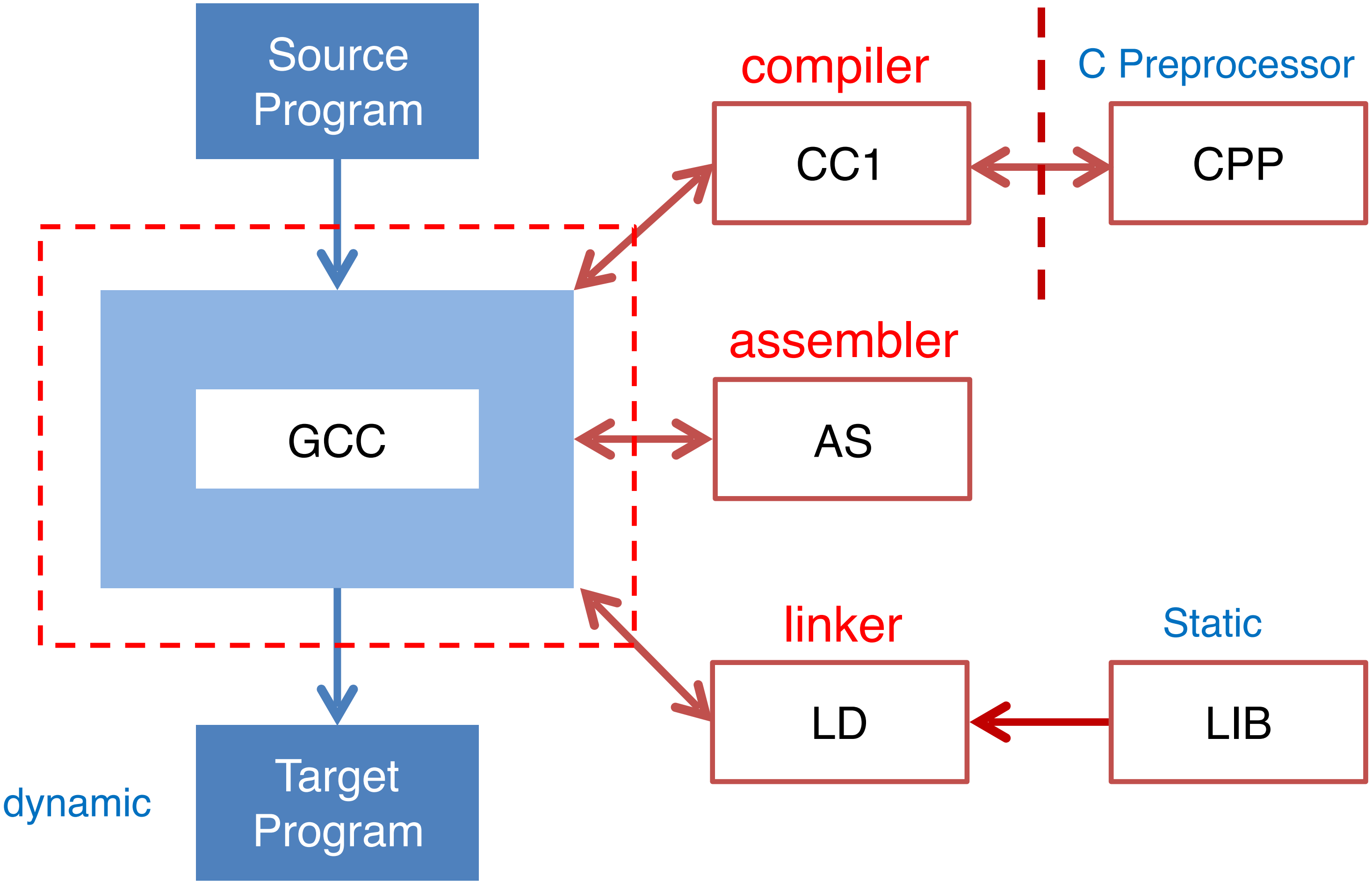
Chapter1

GCC

GNU Compiler Collection

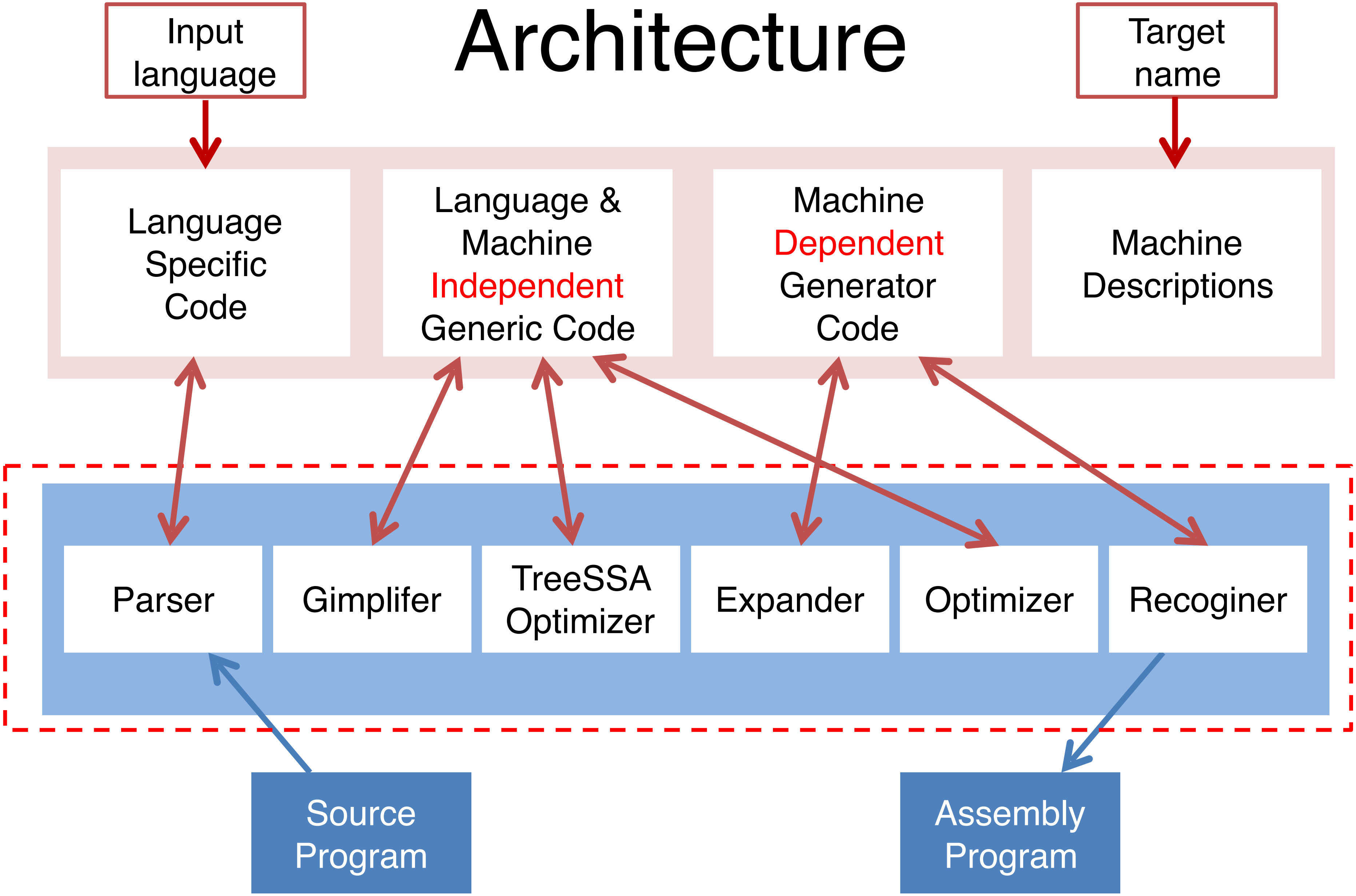
Great Compiler Challenge

GCC compiler

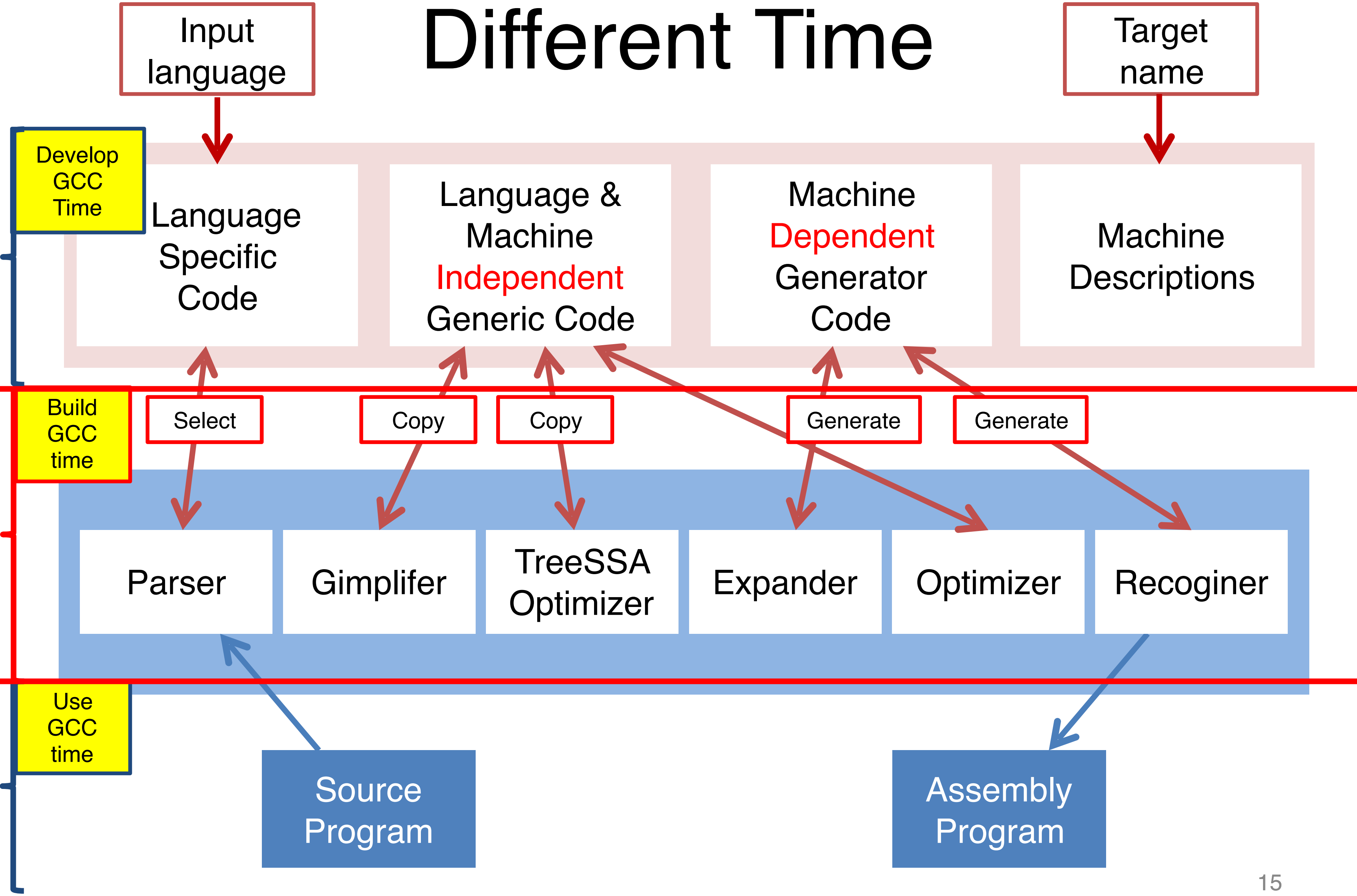


GCC is a collection that invokes compiler, assembler and linker…

Architecture



Different Time



Input language

Target name

Develop GCC Time

Language Specific Code

Language & Machine Independent Generic Code

Machine Dependent Generator Code

Machine Descriptions

Build GCC time

Select

Copy

Copy

Generate

Generate

Parser

Gimplifer

TreeSSA Optimizer

Expander

Optimizer

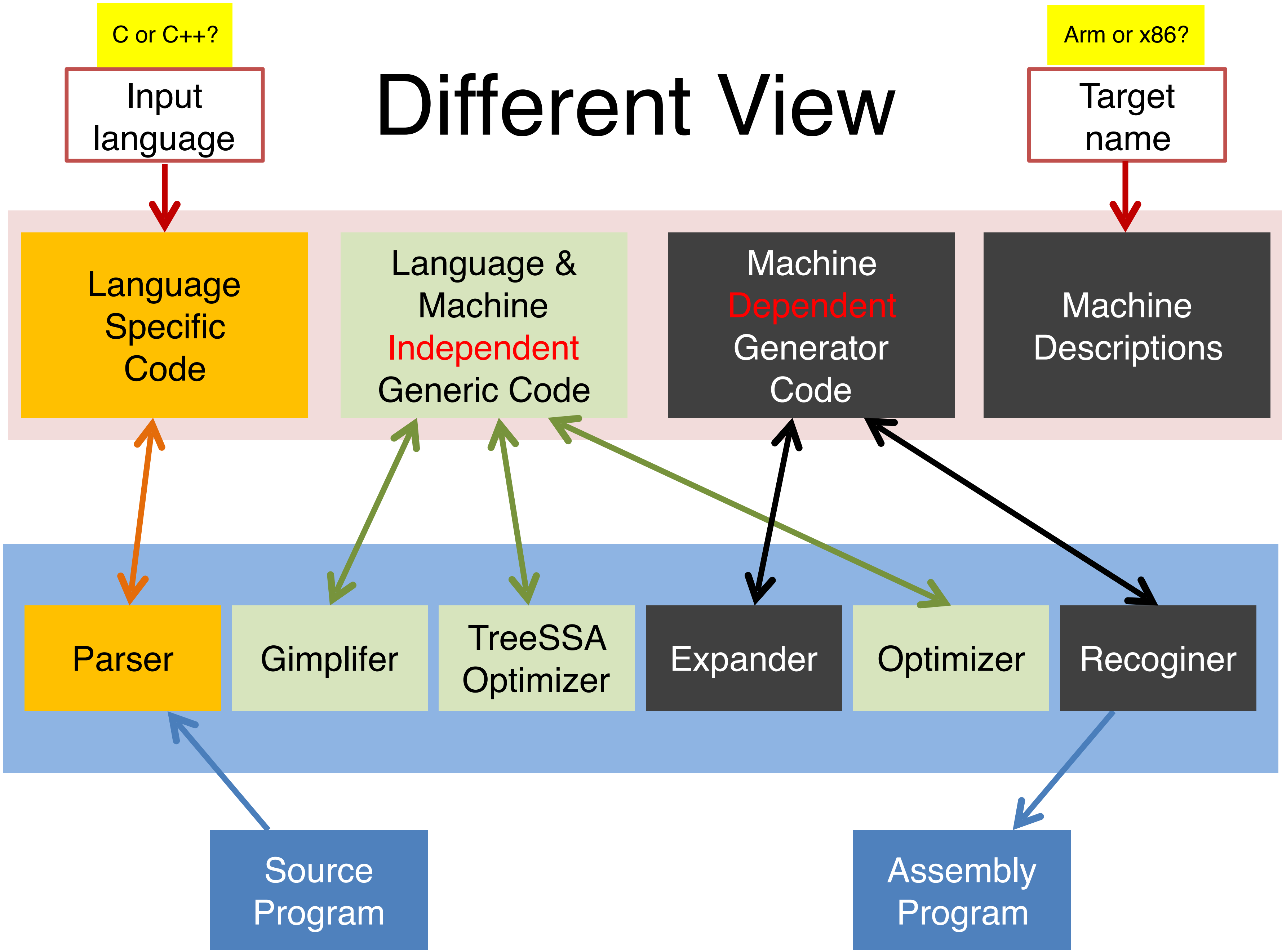
Recoginer

Use GCC time

Source Program

Assembly Program

Different View



C or C++?

Input language

Arm or x86?

Target name

Language Specific Code

Language & Machine Independent Generic Code

Machine Dependent Generator Code

Machine Descriptions

Parser

Gimplifer

TreeSSA Optimizer

Expander

Optimizer

Recoginer

Source Program

Assembly Program

Chapter2

GCC Source Code Configuration & Building

Pre-requisites

- ISO C90
- GCC
- GNU Bash
- Awk
- bzip, gzip, untar
- GNU Make

Mpfr library
Mpc library
Ppl
C LooG-PPL
Jar
Libelf
GMP

https://gcc.gnu.org/install/prerequisites.html

Directory

- GCC Source – source code
  - $(SOURCE_D)
- GCC Build – make source code
  - $(BUILD)
- GCC Install – install binary file
  - $(INSTALL)

*GCC will generate file in build time

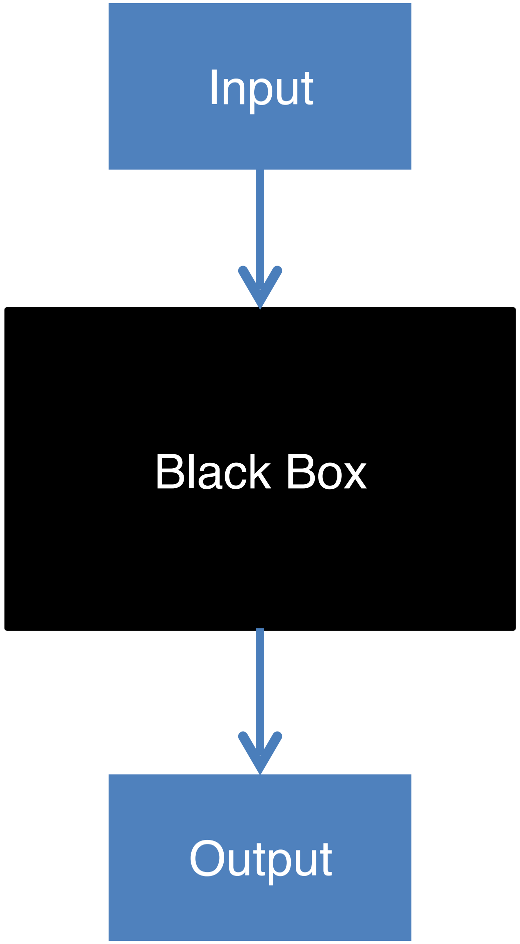
Step

1. Build pre-requisites
2. --prefix = /usr/local => #install path
3. ldconfig => #link library
4. Build gcc

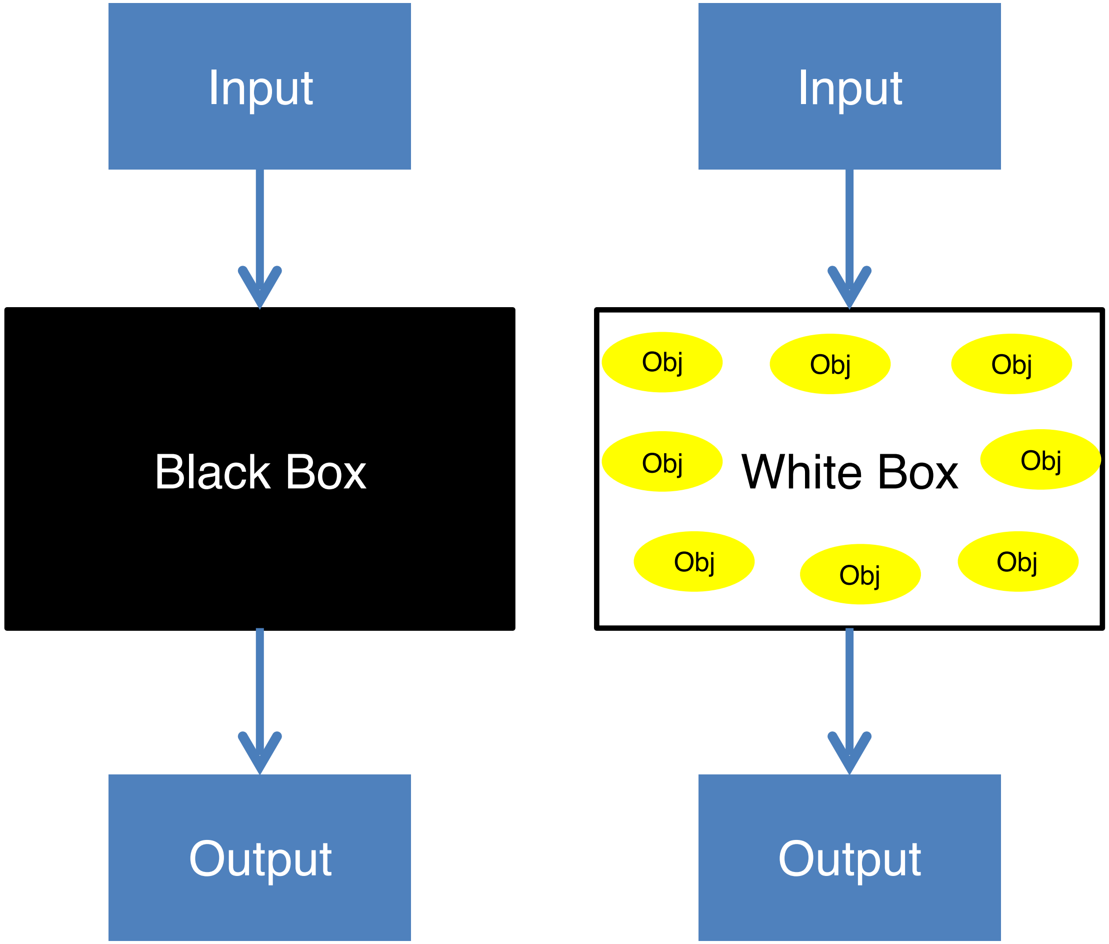
```
cd $(BUILD)
$(SOURCE_D)/configure          => #create makefile
make; make install
```

Chapter3
Gray box probing of GCC

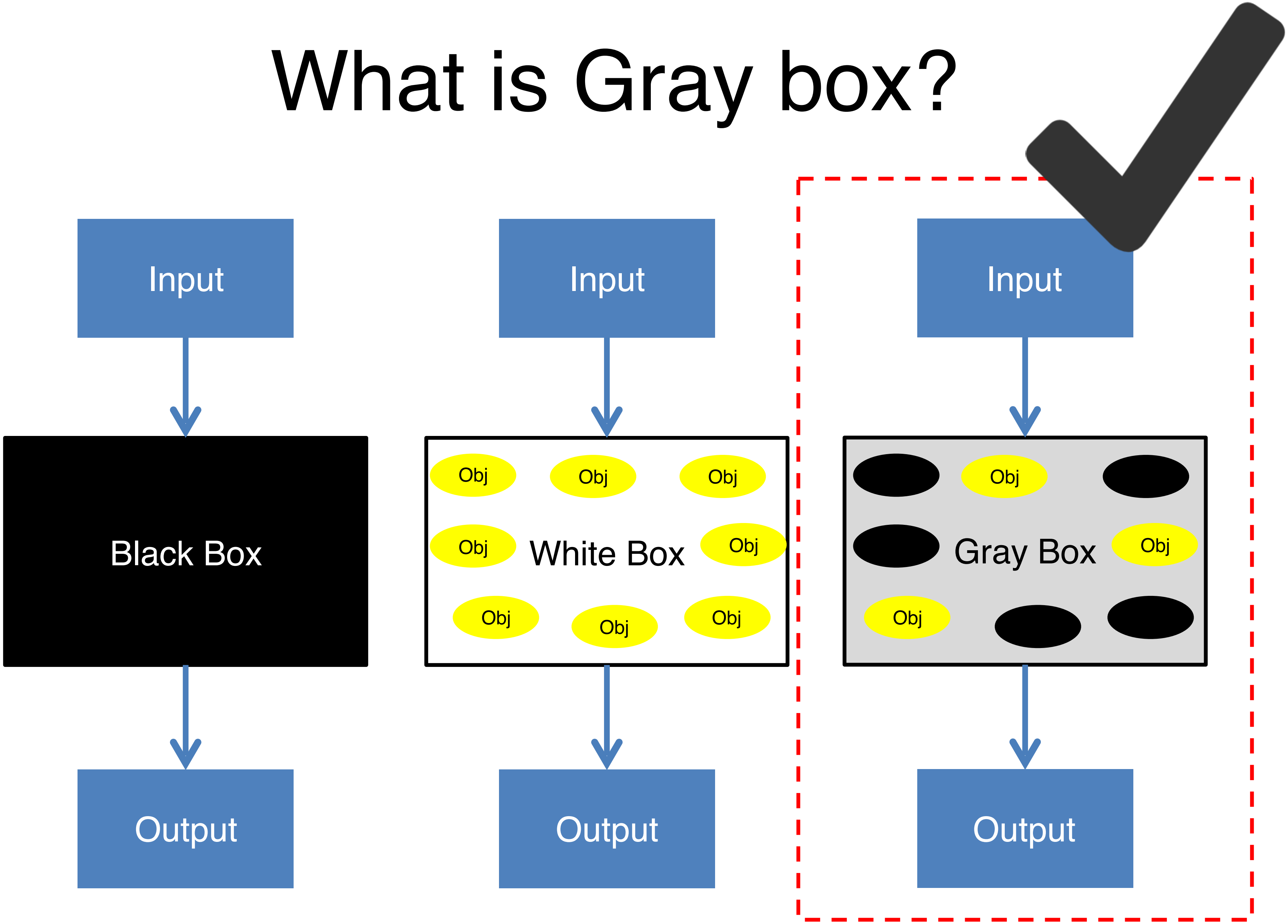
What is Gray box?



What is Gray box?



What is Gray box?

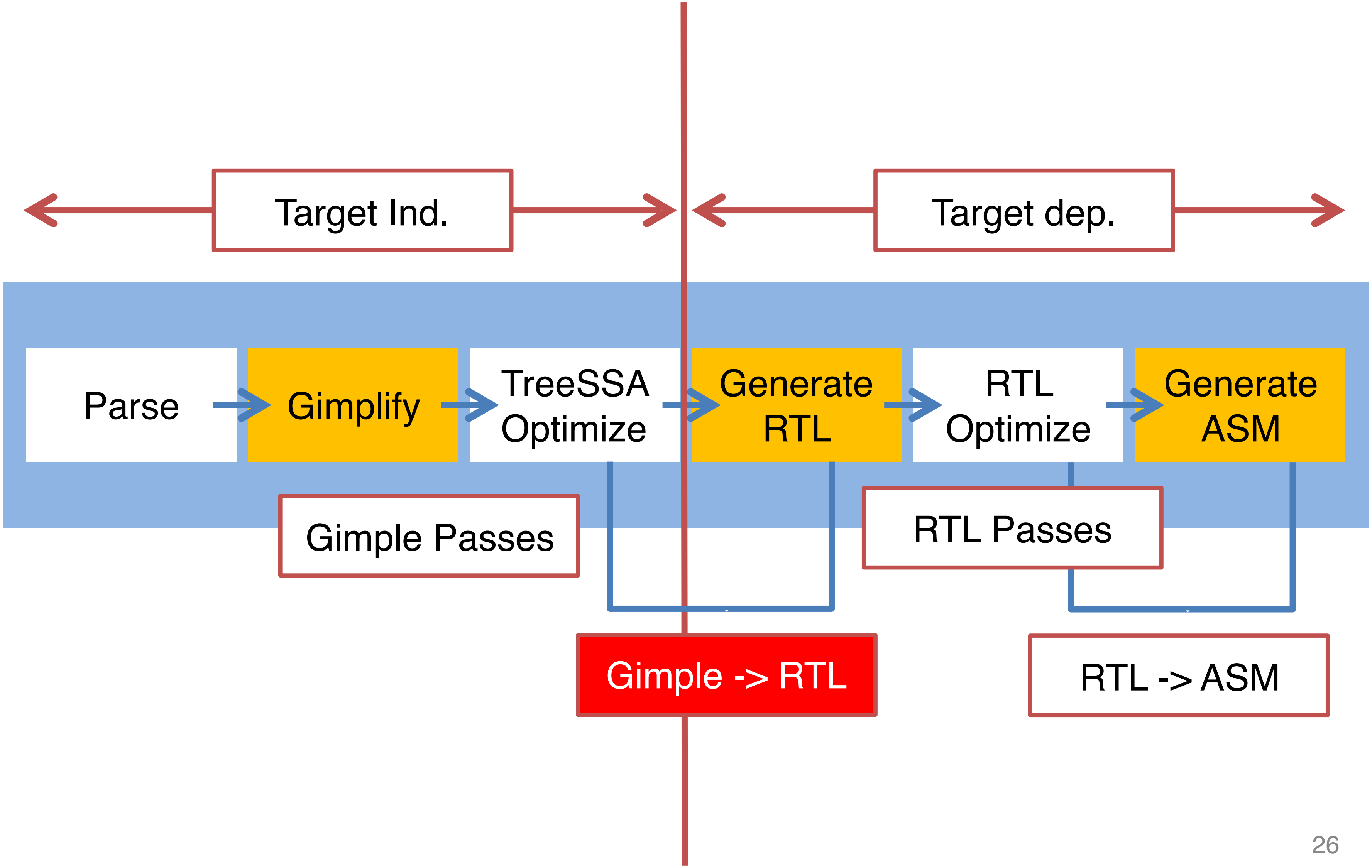


Chapter3

Passes

Examining Dumps

Passes

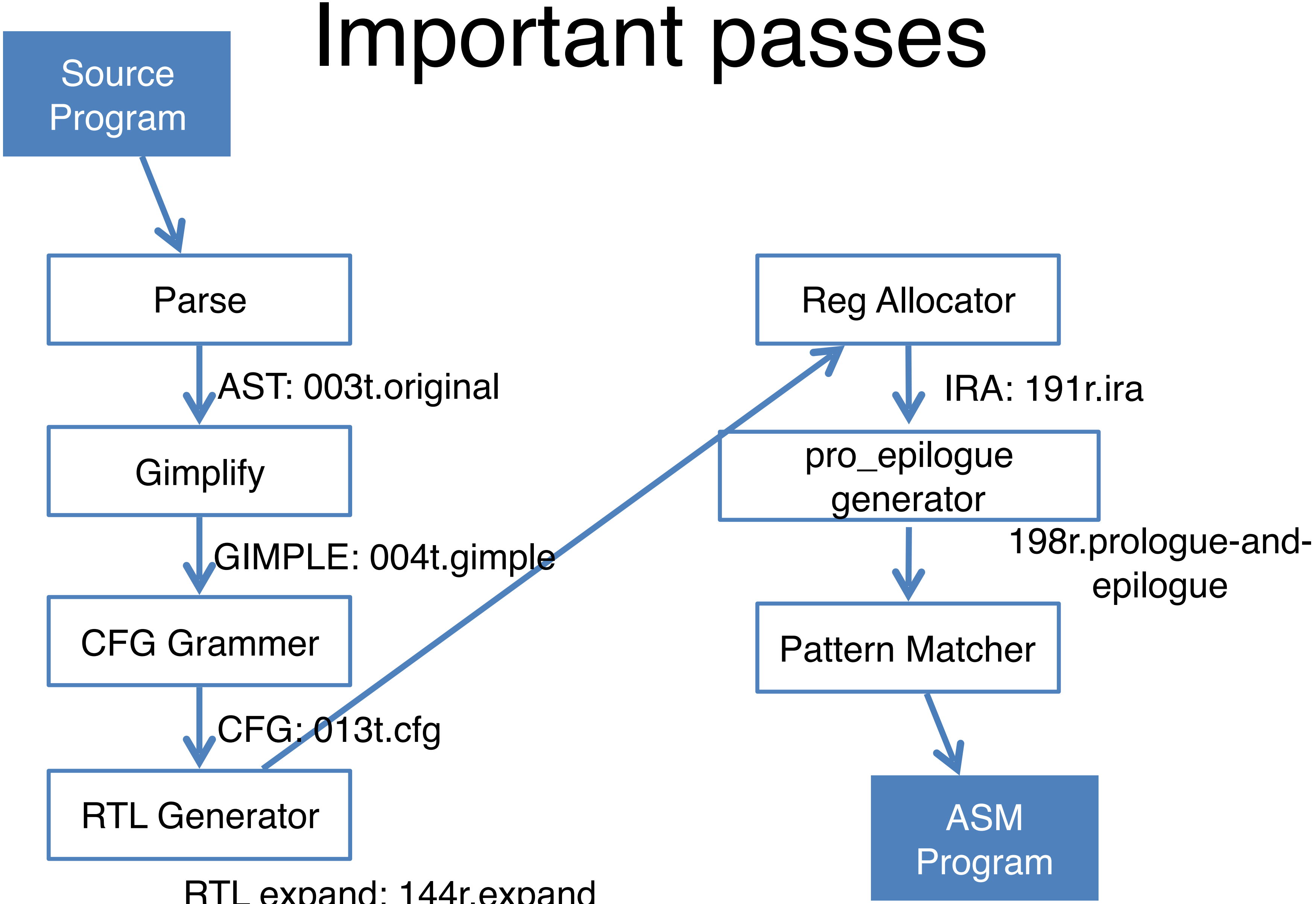

Target Ind.
Target dep.
Parse
Gimplify
TreeSSA Optimize
Generate RTL
RTL Optimize
Generate ASM
Gimple Passes
RTL Passes
Gimple -> RTL
RTL -> ASM

Command

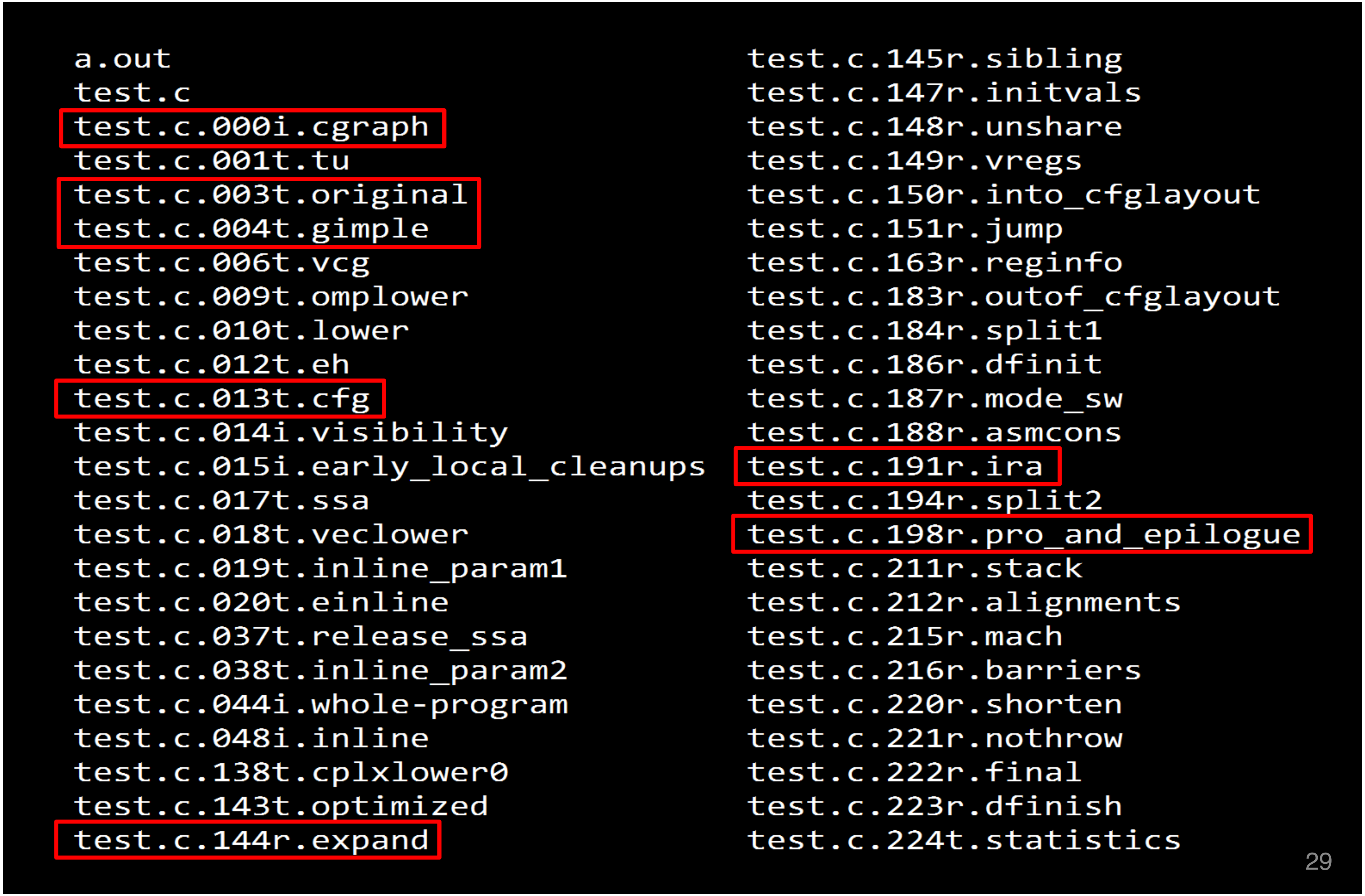
- gcc -fdump-<stage>-<passname> <file>
  - ex. gcc -fdump-tree-original test.c
  - ex. gcc -fdump-tree-cfg-raw test.c
  - ex. gcc -fdump-ipa-all test.c

  - Stage:
    - tree
    - ipa
    - rtl

Important passes



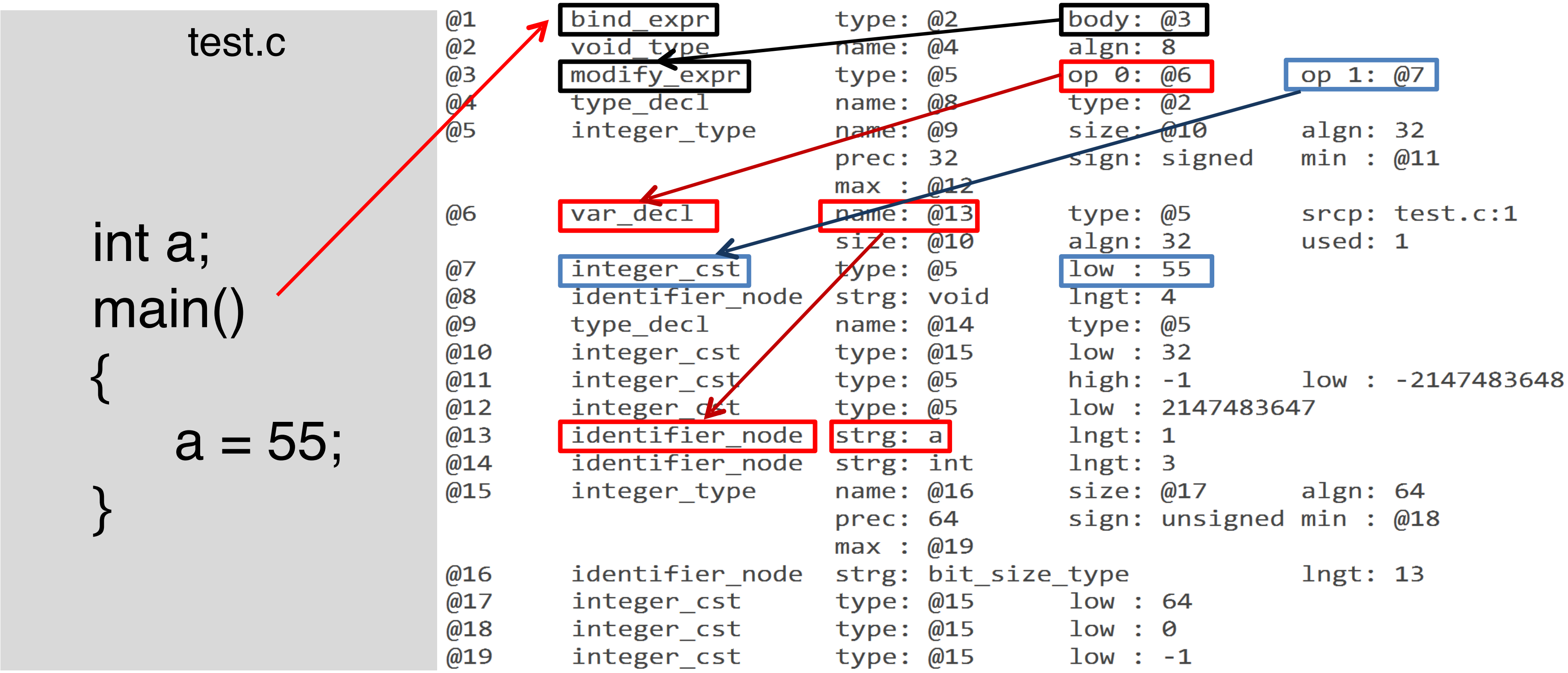
Command Result



```
a.out                             test.c.145r.sibling
test.c                            test.c.147r.initvals
test.c.000i.cgraph                test.c.148r.unshare
test.c.001t.tu                    test.c.149r.vregs
test.c.003t.original              test.c.150r.into_cfglayout
test.c.004t.gimple                test.c.151r.jump
test.c.006t.vcg                   test.c.163r.reginfo
test.c.009t.omplower              test.c.183r.outof_cfglayout
test.c.010t.lower                 test.c.184r.split1
test.c.012t.eh                    test.c.186r.dfinit
test.c.013t.cfg                   test.c.187r.mode_sw
test.c.014i.visibility            test.c.188r.asmcons
test.c.015i.early_local_cleanups  test.c.191r.ira
test.c.017t.ssa                   test.c.194r.split2
test.c.018t.veclower              test.c.198r.pro_and_epilogue
test.c.019t.inline_param1         test.c.211r.stack
test.c.020t.einline               test.c.212r.alignments
test.c.037t.release_ssa           test.c.215r.mach
test.c.038t.inline_param2         test.c.216r.barriers
test.c.044i.whole-program         test.c.220r.shorten
test.c.048i.inline                test.c.221r.nothrow
test.c.138t.cplxlower0            test.c.222r.final
test.c.143t.optimized             test.c.223r.dfinish
test.c.144r.expand                test.c.224t.statistics
```

Examples: AST dumps

1. gcc -fdump-tree-original-raw test.c



test.c

```
int a;
main()
{
      a = 55;
}
```

```
@1      bind_expr        type: @2       body: @3
@2      void_type        name: @4       algn: 8
@3      modify_expr      type: @5       op 0: @6       op 1: @7
@4      type_decl        name: @8       type: @2
@5      integer_type     name: @9       size: @10      algn: 32
                         prec: 32       sign: signed   min : @11
                         max : @12
@6      var_decl         name: @13      type: @5       srcp: test.c:1
                         size: @10      algn: 32       used: 1
@7      integer_cst      type: @5       low : 55
@8      identifier_node  strg: void     lngt: 4
@9      type_decl        name: @14      type: @5
@10     integer_cst      type: @15      low : 32
@11     integer_cst      type: @5       high: -1       low : -2147483648
@12     integer_cst      type: @5       low : 2147483647
@13     identifier_node  strg: a        lngt: 1
@14     identifier_node  strg: int      lngt: 3
@15     integer_type     name: @16      size: @17      algn: 64
                         prec: 64       sign: unsigned min : @18
                         max : @19
@16     identifier_node  strg: bit_size_type          lngt: 13
@17     integer_cst      type: @15      low : 64
@18     integer_cst      type: @15      low : 0
@19     integer_cst      type: @15      low : -1
```

Examples: GIMPLE dumps

2. gcc -fdump-tree-gimple test.c

test.c

```
int main()
{
        int a[3], x;
        a[1] = a[2] = 10;
         x = a[1] + a[2];
         a[0] = a[1] + a[1]*x;
}
```

test.c.004t.gimple

```
main ()
{
  int D.1589;
  int D.1590;
  int D.1591;
  int D.1592;
  int D.1593;
  int D.1594;
  int a[3];
  int x;
a[2] = 10;
  D.1589 = a[2];
  a[1] = D.1589;
  D.1590 = a[1];
  D.1591 = a[2];
  x = D.1590 + D.1591;
  D.1592 = x + 1;
  D.1593 = a[1];
  D.1594 = D.1592 * D.1593;
  a[0] = D.1594;
}
```

Examples: CFG dumps

3. gcc -fdump-tree-cfg test.c

test.c (part)

```
If (a<=12)
        a = a+b+c;
```

test.c.004t.gimple (part)

```
if (a<=12) goto
<D.1200>
else goto <D.1201>
<D.1200>:
D.1199 = a + b;
a = D.1199 + c;
<D.1201>:
```

Examples: RTL dumps

4. gcc -fdump-rtl-expand test.c

test.c

```
int a;
main()
{
    a = a+1;
}
```

test.c.144r.expand (part)

```
(insn 5 4 6 3 (set (reg:SI 59 [ a.0 ])
        (mem/c/i:SI (symbol_ref:DI ("a")  <var_decl
0x7f13bdac3000 a>) [0 a+0 S4
 A32])) test.c:4 -1
     (nil))

(insn 6 5 7 3 (parallel [
            (set (reg:SI 60 [ a.1 ])
                (plus:SI (reg:SI 59 [ a.0 ])
                    (const_int 1 [0x1])))
            (clobber (reg:CC 17 flags))
        ]) test.c:4 -1
     (nil))

(insn 7 6 13 3 (set (mem/c/i:SI (symbol_ref:DI ("a")
<var_decl 0x7f13bdac3000 a>) [0 a+0 S4 A32])
        (reg:SI 60 [ a.1 ])) test.c:4 -1
     (nil))
```

r59 = a;

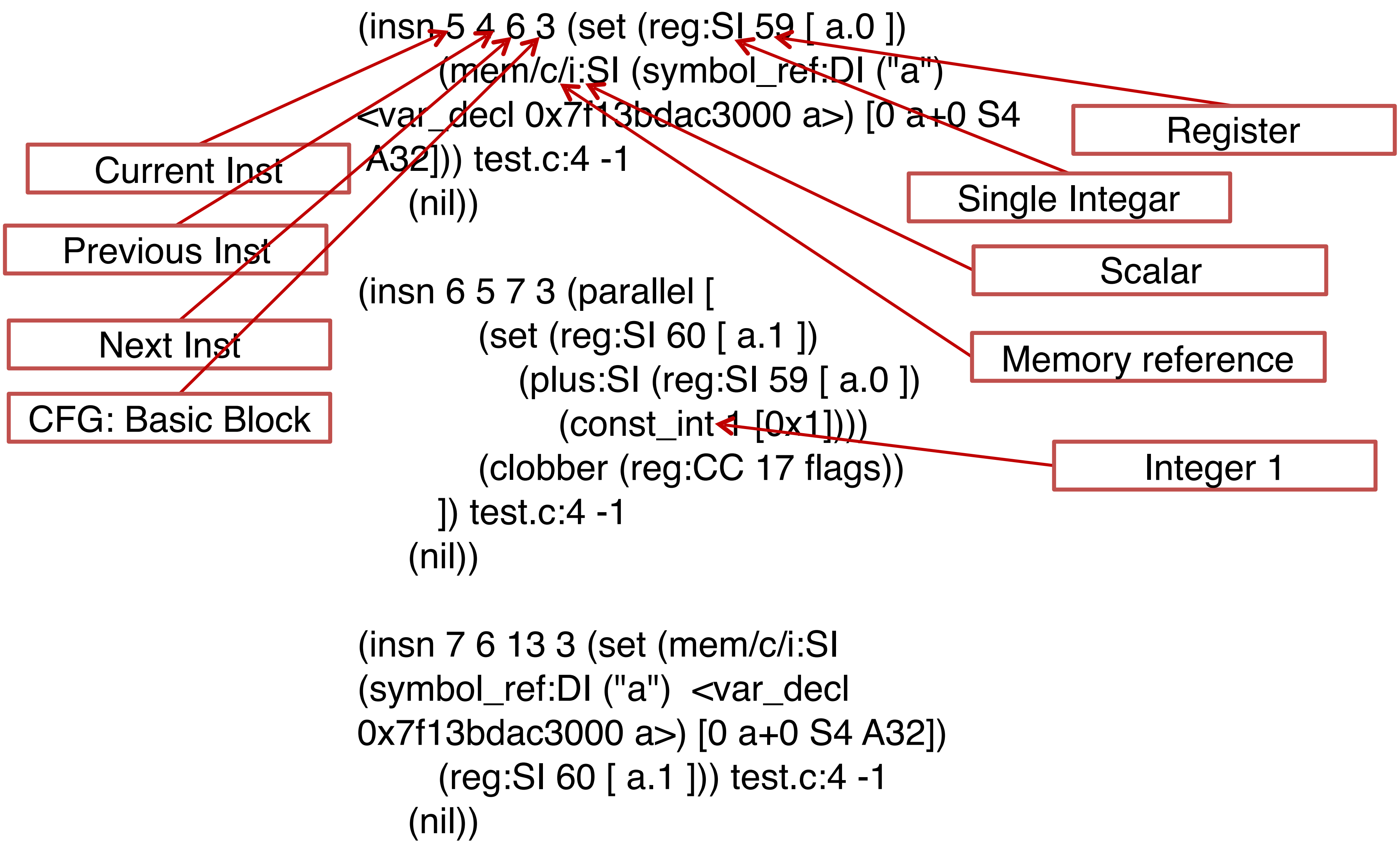
```
(insn 5 4 6 3 (set (reg:SI 59 [ a.0 ])
        (mem/c/i:SI (symbol_ref:DI ("a")
<var_decl 0x7f13bdac3000 a>) [0 a+0 S4
 A32])) test.c:4 -1
     (nil))
```

r60 = r59 + 1

```
(insn 6 5 7 3 (parallel [
            (set (reg:SI 60 [ a.1 ])
                (plus:SI (reg:SI 59 [ a.0 ])
                    (const_int 1 [0x1])))
            (clobber (reg:CC 17 flags))
        ]) test.c:4 -1
     (nil))
```

a = r60

```
(insn 7 6 13 3 (set (mem/c/i:SI
(symbol_ref:DI ("a")  <var_decl
0x7f13bdac3000 a>) [0 a+0 S4 A32])
        (reg:SI 60 [ a.1 ])) test.c:4 -1
     (nil))
```



```
(insn 5 4 6 3 (set (reg:SI 59 [ a.0 ])
       (mem/c/i:SI (symbol_ref:DI ("a")
<var_decl 0x7f13bdac3000 a>) [0 a+0 S4
 A32])) test.c:4 -1
     (nil))

(insn 6 5 7 3 (parallel [
           (set (reg:SI 60 [ a.1 ])
               (plus:SI (reg:SI 59 [ a.0 ])
                   (const_int 1 [0x1])))
           (clobber (reg:CC 17 flags))
       ]) test.c:4 -1
     (nil))

(insn 7 6 13 3 (set (mem/c/i:SI
(symbol_ref:DI ("a")  <var_decl
0x7f13bdac3000 a>) [0 a+0 S4 A32])
       (reg:SI 60 [ a.1 ])) test.c:4 -1
     (nil))
```

Current Inst

Previous Inst

Next Inst

CFG: Basic Block

Register

Single Integar

Scalar

Memory reference

Integer 1

Examples: Assembly dumps

5. gcc -S test.c | | objdump -d a.out

test.c

```
int main()
{
int a;
a=1;
}
```

test.c.144r.expand (part)

```
(insn 5 4 11 3 (set (mem/c/i:SI (plus:DI (reg/f:DI
54 virtual-stack-vars)
                (const_int -4 [0xfffffffffffffffc])) [0 a+0 S4
A32])
        (const_int 1 [0x1])) test.c:4 -1
     (nil))
```

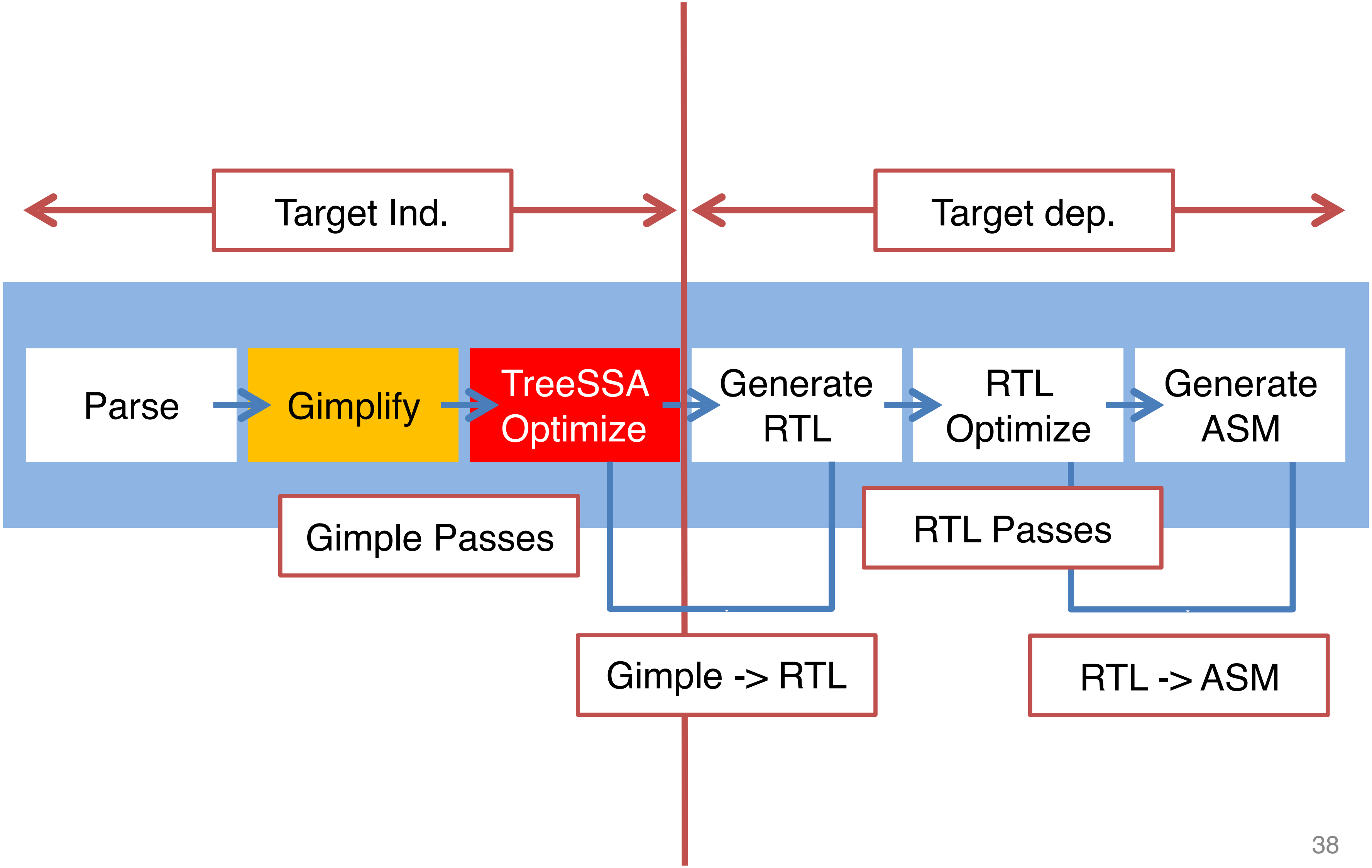
test.s (part)

```
main:
.LFB0:
    .cfi_startproc
    pushq   %rbp
    .cfi_def_cfa_offset 16
    .cfi_offset 6, -16
    movq    %rsp, %rbp
    .cfi_def_cfa_register 6
    movl    $1, -4(%rbp)
    popq    %rbp
    .cfi_def_cfa 7, 8
    ret
    .cfi_endproc
```

Chapter3

Gimple Optimization

Passes



Target Ind.
Target dep.
Parse
Gimplify
TreeSSA Optimize
Generate RTL
RTL Optimize
Generate ASM
Gimple Passes
RTL Passes
Gimple -> RTL
RTL -> ASM

Brief

0. Pre-procedure (017t.ssa) (022t.copyrename1)
1. constant propagation (023t.ccp1) (059t.ccp2)
2. copy propagation (027t.copyprop1)
3. loop unrolling (058t.cunrolli)
4. dead code elimination (029t.cddce1)

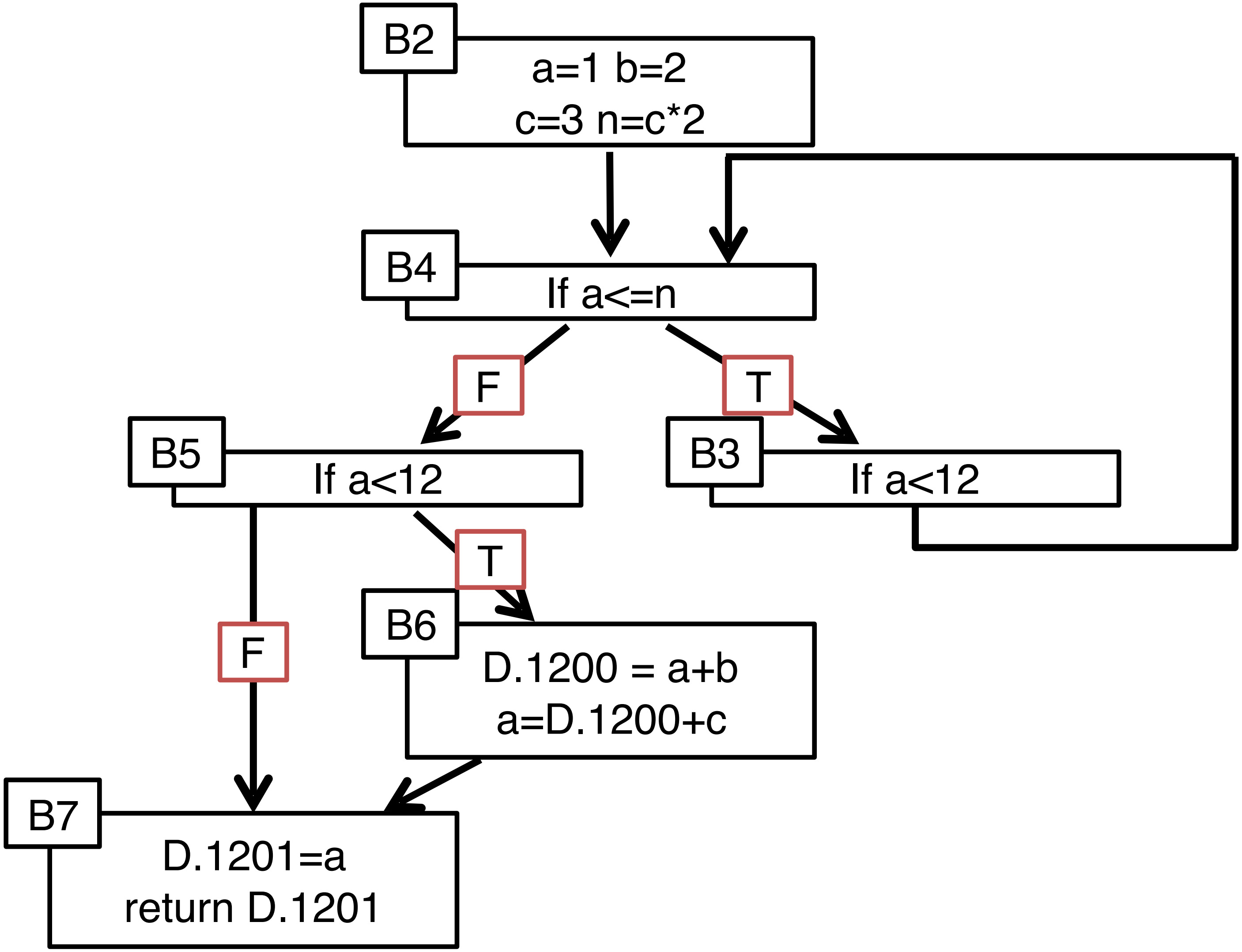
Command:

```
gcc -fdump-tree-all -O2 test.c
```

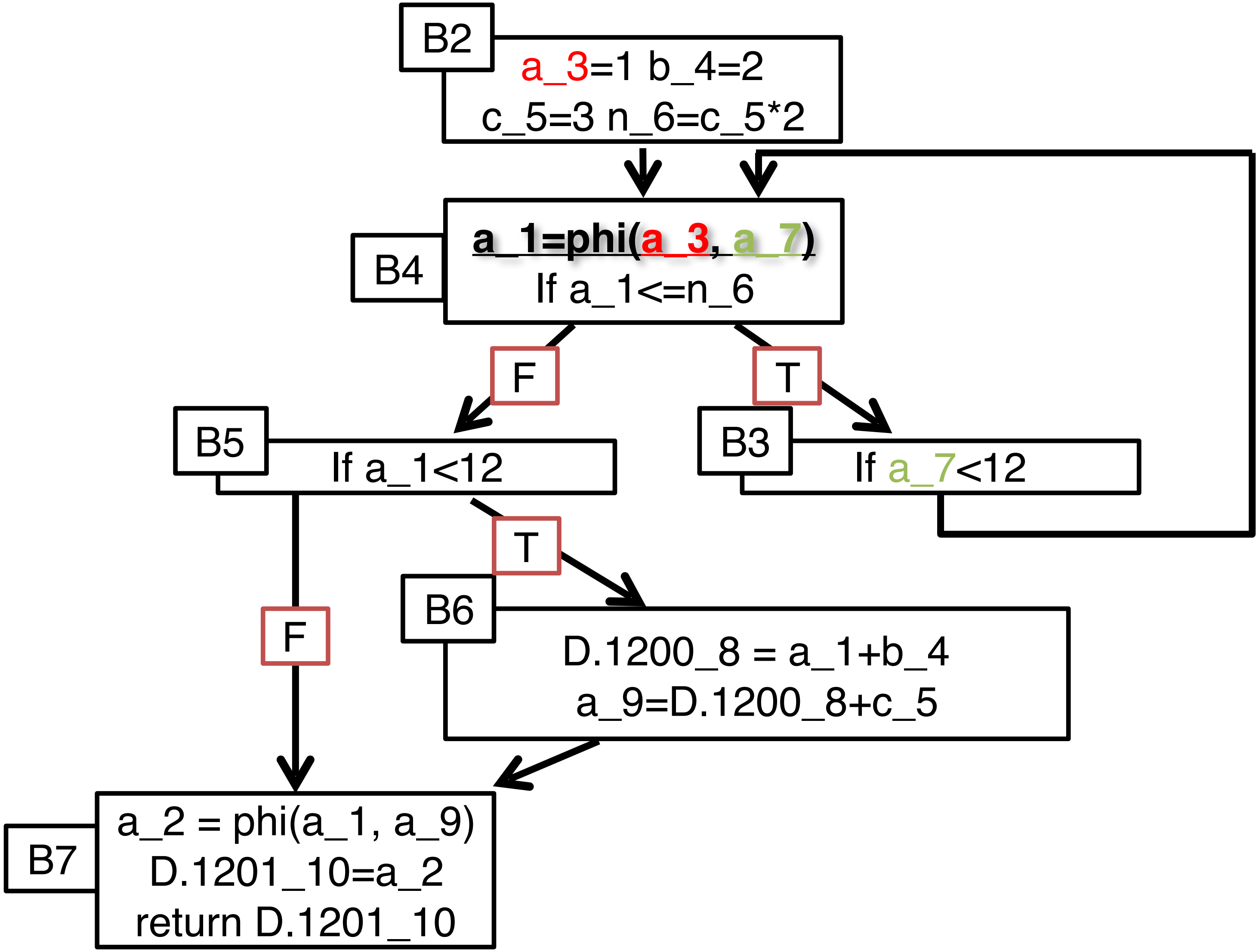
Source Code

```
int main()
{
        int a=1;
        int b=2;
        int c=3;
        int n=c*2;
        while (a<=n)
                a = a+1;
        if (a<12)
                a = a+b+c;
        return a;
}
```

cfg (Control Flow Graph)



cfg -> ssa (017t.ssa)



ssa -> copyrename (022t)

017t.ssa

```
<bb 7>:
#a_2 = PHI<a_1(5),a_9(6)>
D.1201_10=a_2;
return D.1201_10;
```

022t.copyrename1

```
<bb 7>:
#a_2 = PHI<a_1(5),a_9(6)>
a_10=a_2;
return a_10;
```

copyrename -> ccp (023t)

022t.copyrename1

```
<bb 2>
a_3=1;
b_4=2;
c_5=3;
n_6=c_5*2;
goto <bb 4>
```

023t.ccp1

```
<bb 2>
a_3=1;
b_4=2;
c_5=3;
n_6=6;
goto <bb 4>
```

ccp -> copyprop (027t)

023t.ccp1

```
<bb 7>:
#a_2 = PHI<a_1(5),a_9(6)>
~~a_10=a_2;~~
return a_10;
```

027t.copyprop1

```
<bb 7>:
#a_2 = PHI<a_1(5),a_9(6)>
return a_2;
```

copyprop -> cddc (029t)

029t.cddc

<bb 2>:
~~a_3 = 1;~~
~~b_4 = 2;~~
~~c_5 = 3;~~
~~n_6 = 6;~~
goto <bb 4>;

<bb 3>:
a_7 = a_1 + 1;

<bb 4>:
…

029t.cddc

```
<bb 2>:
goto <bb 4>;

<bb 3>:
a_7 = a_1 + 1;

<bb 4>:
…
```

cddc ->cunrolli (058t)

029t.cddc

```
<bb 2>:
goto <bb 4>;

<bb 3>:
a_7 = a_1 + 1;

<bb 4>:
# a_1 = PHI <1(2), a 7(3)>
if (a_1 <= 6) goto <bb 3>;
else goto <bb 5>;

<bb 5>:
if (a_1 <= 11) goto <bb 6>;
else goto <bb 7>;

<bb 6>:
a_9 = a_1 + 5;

<bb 7>:
# a_2 = PHI <a_1(5), a_9(6)>
return a_2;
```

058t.cunrolli

```
<bb 2>:
a_12 = 2;
a_14 = a_12 + 1;
a_16 = a_14 + 1;
a_18 = a_16 + 1;
a_20 = a_18 + 1;
a_22 = a_20 + 1;
if (a_22 <= 11) goto <bb 3>;
else goto <bb 4>;

<bb 3>:
a_9 = a_22 + 5;

<bb 4>:
# a_2 = PHI <a_22(2), a_9(3)>
return a_2;
```

cunrolli -> ccp2 (059t)

058t.cunrolli

```
<bb 2>:
a_12 = 2;
a_14 = a_12 + 1;
a_16 = a_14 + 1;
a_18 = a_16 + 1;
a_20 = a_18 + 1;
a_22 = a_20 + 1;
if (a_22 <= 11) goto <bb 3>;
else goto <bb 4>;

<bb 3>:
a_9 = a_22 + 5;

<bb 4>:
# a_2 = PHI <a_22(2), a_9(3)>
return a_2;
```

059t.ccp2

```
main()
{
<bb 2>:
        return 12;
}
```

Reference

- GCC Source Code
  - https://github.com/mirrors/gcc
- IITB GCC workshop OCW
  - http://www.cse.iitb.ac.in/grc/index.php?page=gcc-pldi14-tut