intelligence everywhere™

Guidelines for Using SDL in Product Development

Frank Weil

Motorola

Global Software Group – Software Design Automation

04 June 2004

MOTOROLA

Overview

- **General Recommendations for Design Models**
- **Recommendations on SDL Usage for Design**
- **Performance Considerations**
- **Platform Interfaces**
- **Portability Issues**
- **Example Models**

SDL for Design

Two inherent problems:

- **Bad things can happen behind the scenes**
  - **Deadlocks**
  - **Signal send to a dead / invalid / unreachable process**
  - **…**
  - **It is hard to *prove* correctness!**
- **Syntactically/semantically correct models can be confusing**
  - **Subrange constants are ignored in data type definition**
  - **Unused signals, data types, variables, procedures, …**
  - **…**
  - **Excuse: “If they new the semantics better, it would not be confusing”**
  - **Obvious things should have obvious semantics**

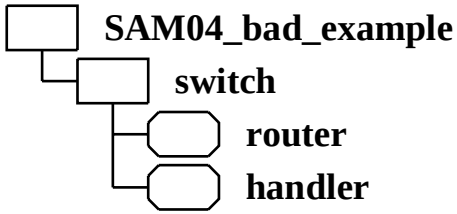
General Recommendations

- **Nondeterminism and Fairness**
  - **SDL has several nondeterministic features**
  - **A good design should be deterministic**
  - ***Nondeterminism does not imply fairness***

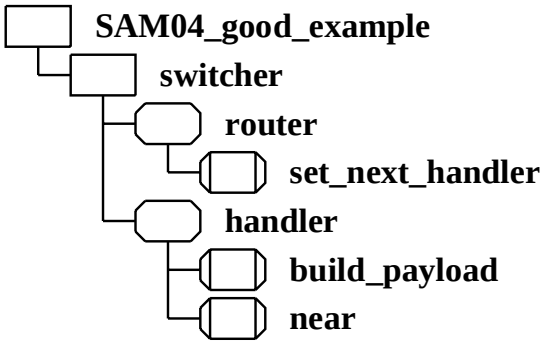
- **Abstraction**
  - **Everything should be made as simple as possible, but no simpler! - *Albert Einstein***
  - **Model must capture concepts, not irrelevant details**

Examples Models

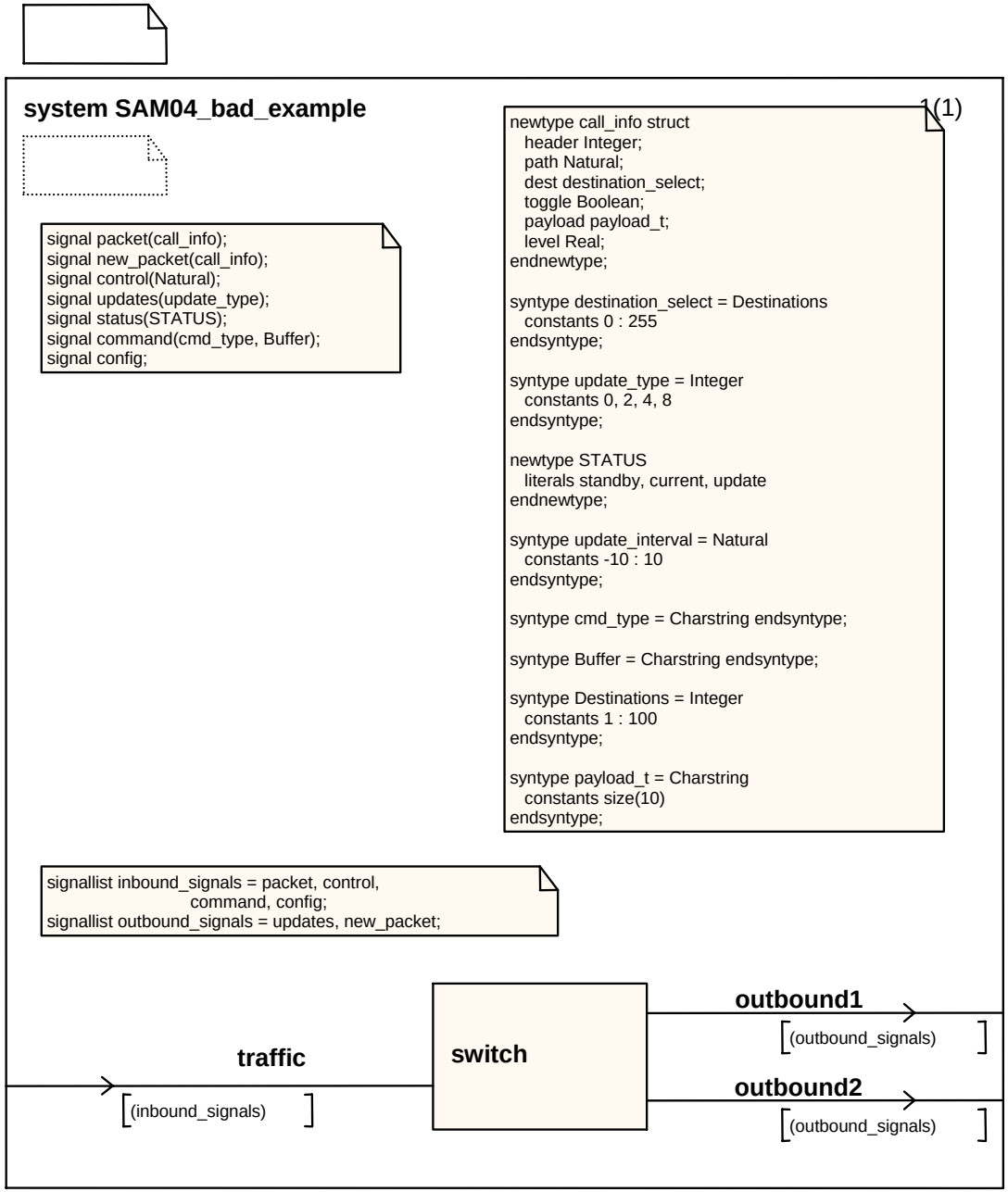
- **First, a bad example**



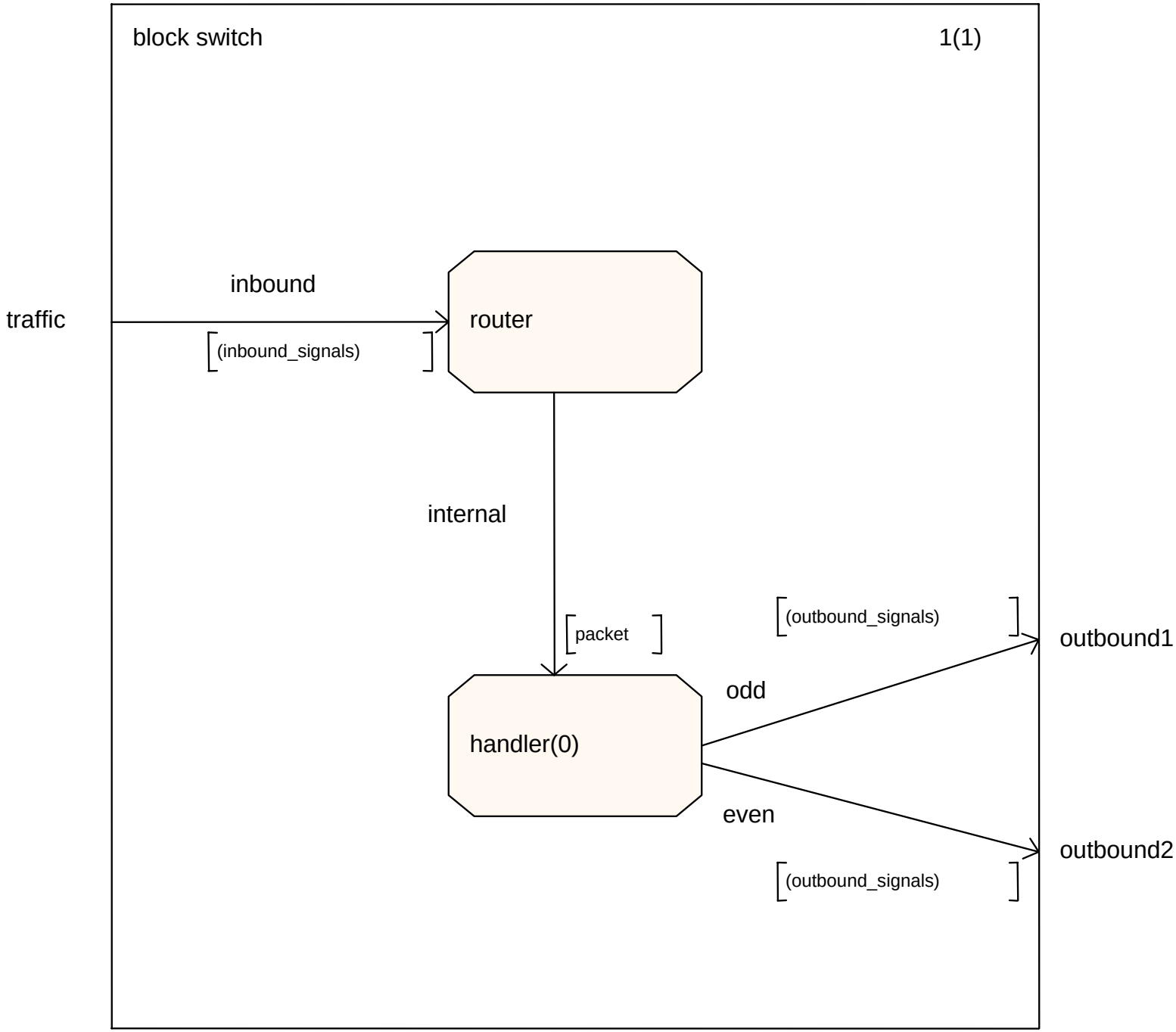
- **Then, a good example for comparison**



- **Read the paper for details (*please?*)**


system SAM04_bad_example
1 (1)
signal packet(call_info);
signal new_packet(call_info);
signal control(Natural);
signal updates(update_type);
signal status(STATUS);
signal command(cmd_type, Buffer);
signal config;
newtype call_info struct
header Integer;
path Natural;
dest destination_select;
toggle Boolean;
payload payload_t;
level Real;
endnewtype;
syntype destination_select = Destinations
constants 0 : 255
endsyntype;
syntype update_type = Integer
constants 0, 2, 4, 8
endsyntype;
newtype STATUS
literals standby, current, update
endnewtype;
syntype update_interval = Natural
constants -10 : 10
endsyntype;
syntype cmd_type = Charstring endsyntype;
syntype Buffer = Charstring endsyntype;
syntype Destinations = Integer
constants 1 : 100
endsyntype;
syntype payload_t = Charstring
constants size(10)
endsyntype;
signallist inbound_signals = packet, control,
command, config;
signallist outbound_signals = updates, new_packet;
traffic
(inbound_signals)
switch
outbound1
(outbound_signals)
outbound2
(outbound_signals)

Frank Weil, Frank.Weil@motorola.com



process router 1(2)

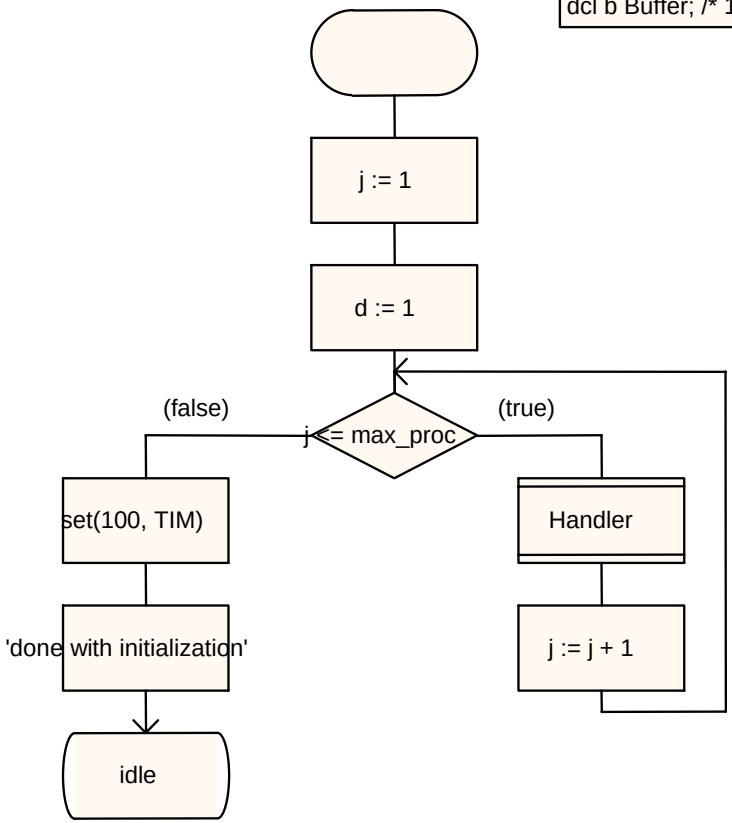
```
synonym 911 cmd_type = 'EMERGENCY';
synonym REMOVE_CMD cmd_type = 'REMOVE';
synonym max_proc processes = 5;
```

```
procedure log;
fpar in stat Integer;
external;
```

```
syntype processes = Integer
   constants 1 : 5
endsyntype;
```

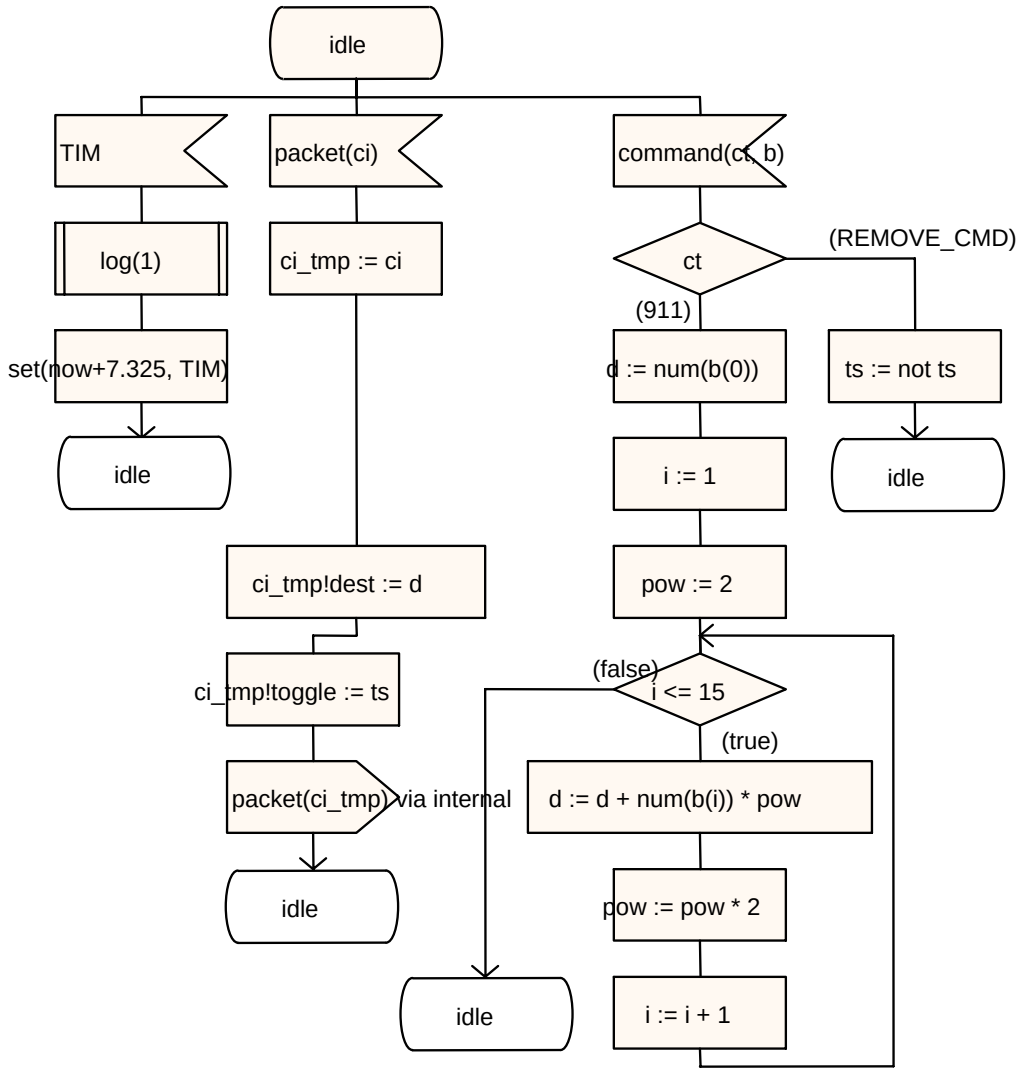
```
timer TIM;
```

```
dcl i, d, pow Integer;
dcl j processes;
dcl ci, ci_tmp call_info;
dcl ct cmd_type;
dcl ts Boolean := false;
dcl b Buffer; /* 16-bit unsigned int, LSB first */
```

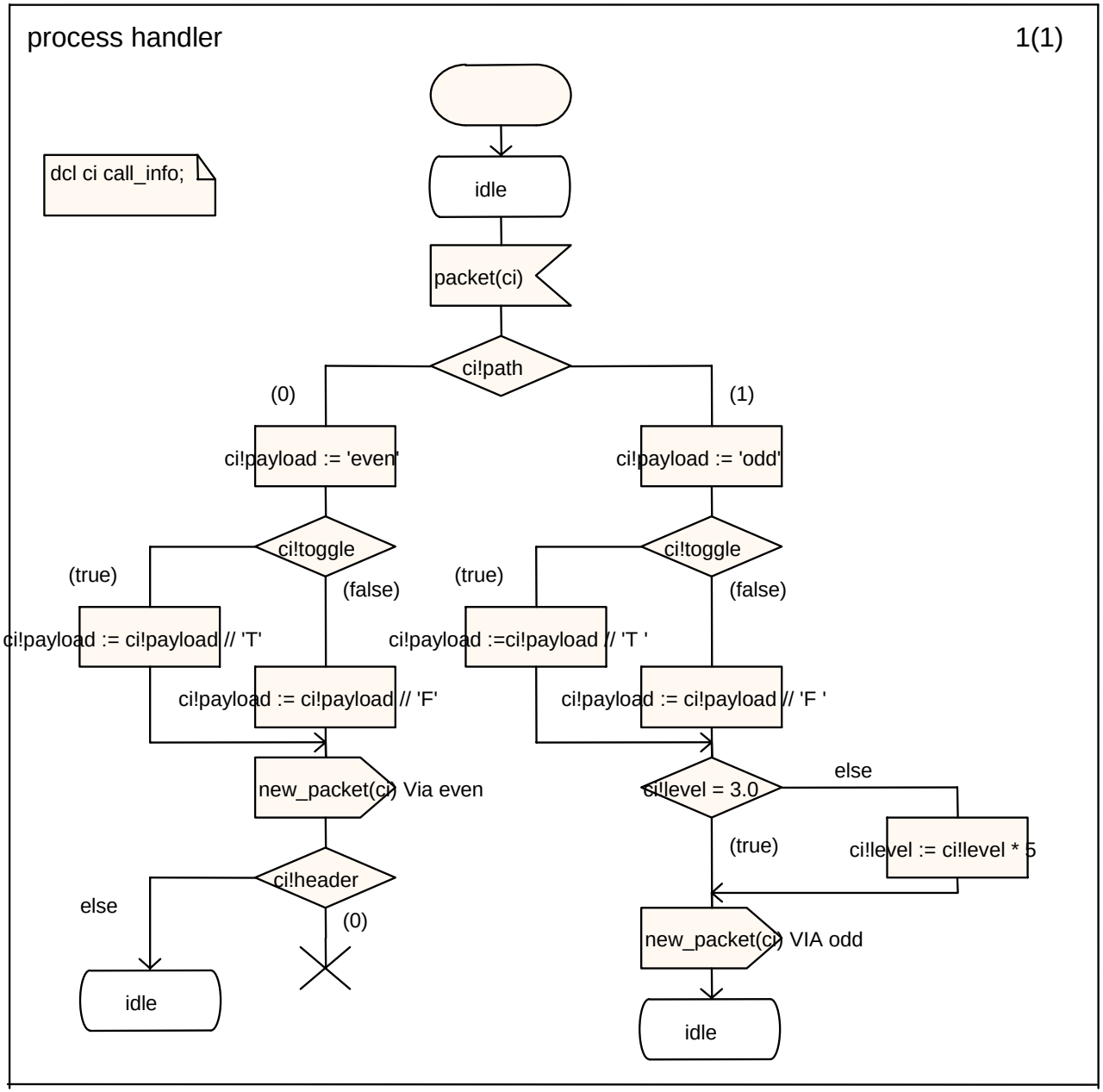


process router

2(2)



Frank Weil, Frank.Weil@motorola.com



process router

1(2)

number as identifier

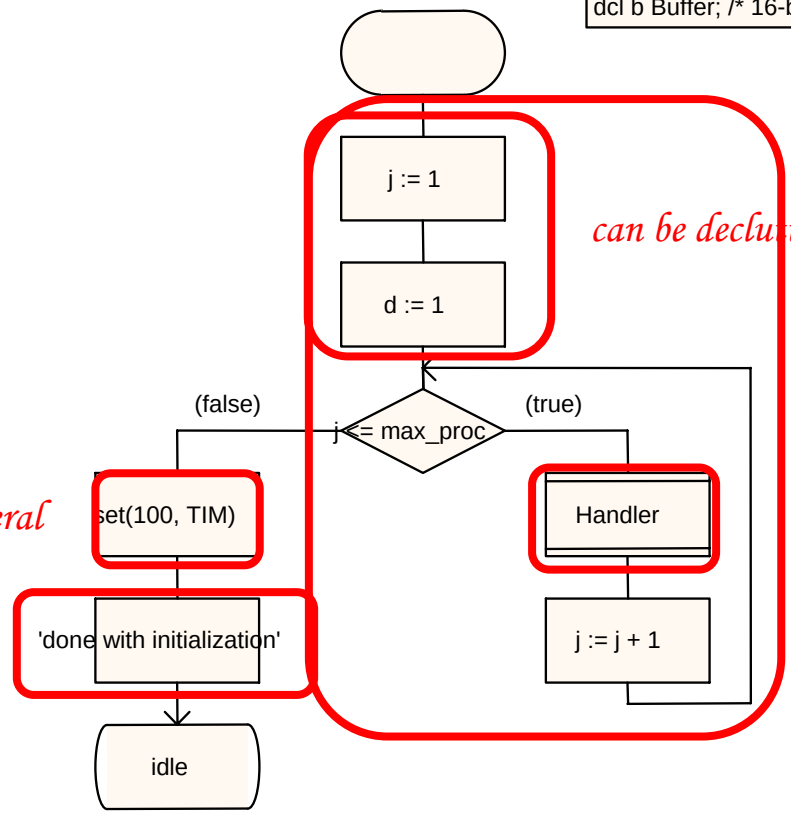
Charstring instead of enum

```
synonym 911 cmd_type = 'EMERGENCY';
synonym REMOVE_CMD cmd_type = 'REMOVE';
synonym max_proc processes = 5;
```

```
syntype processes = Integer
   constants 1 : 5
endsyntype;
```

```
timer TIM;
```

```
dcl i, d, pow Integer;
dcl j processes;
dcl ci, ci_tmp call_info;
dcl ct cmd_type;
dcl ts Boolean := false;
dcl b Buffer; /* 16-bit unsigned int, LSB first */
```



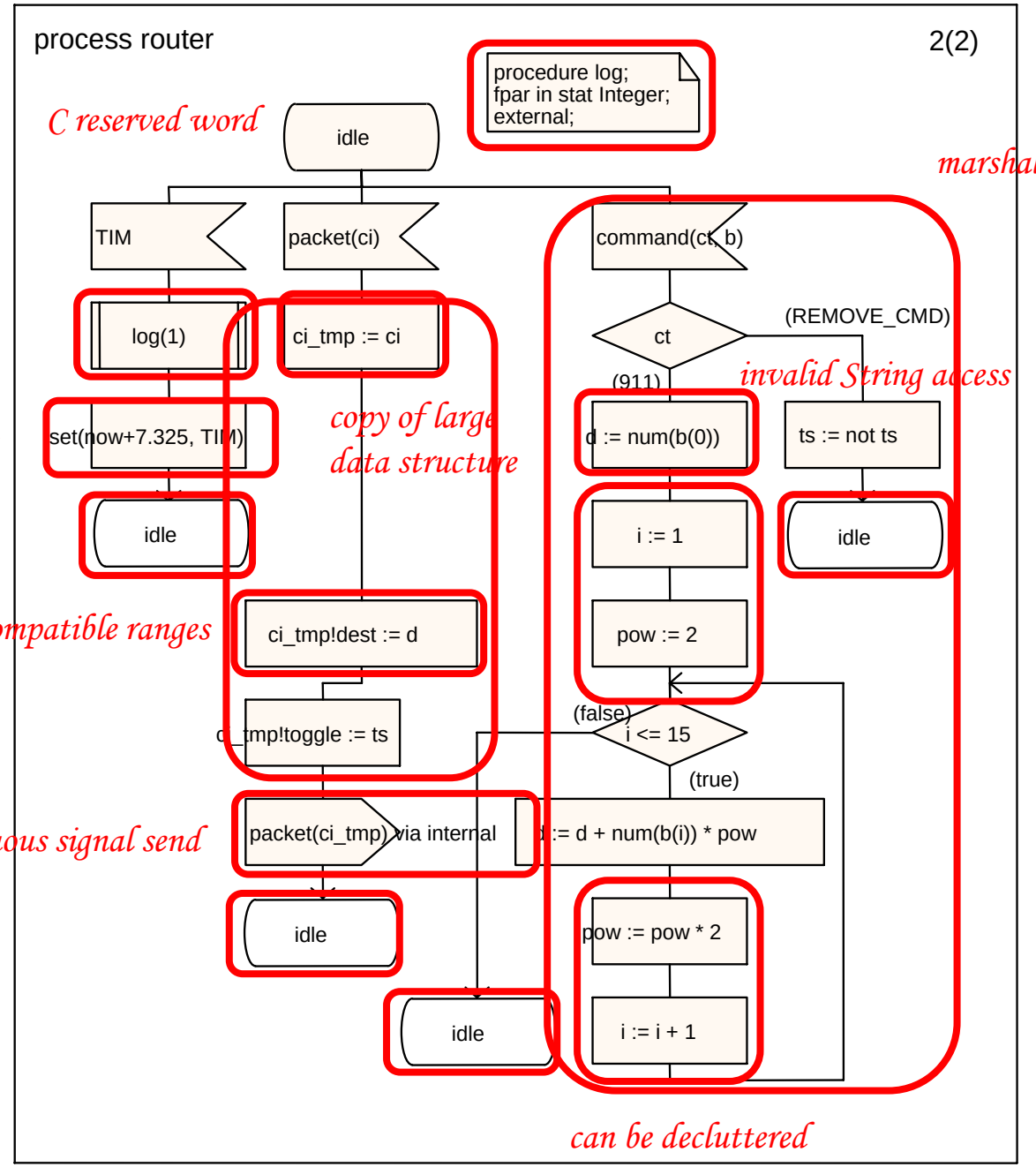
can be declustered

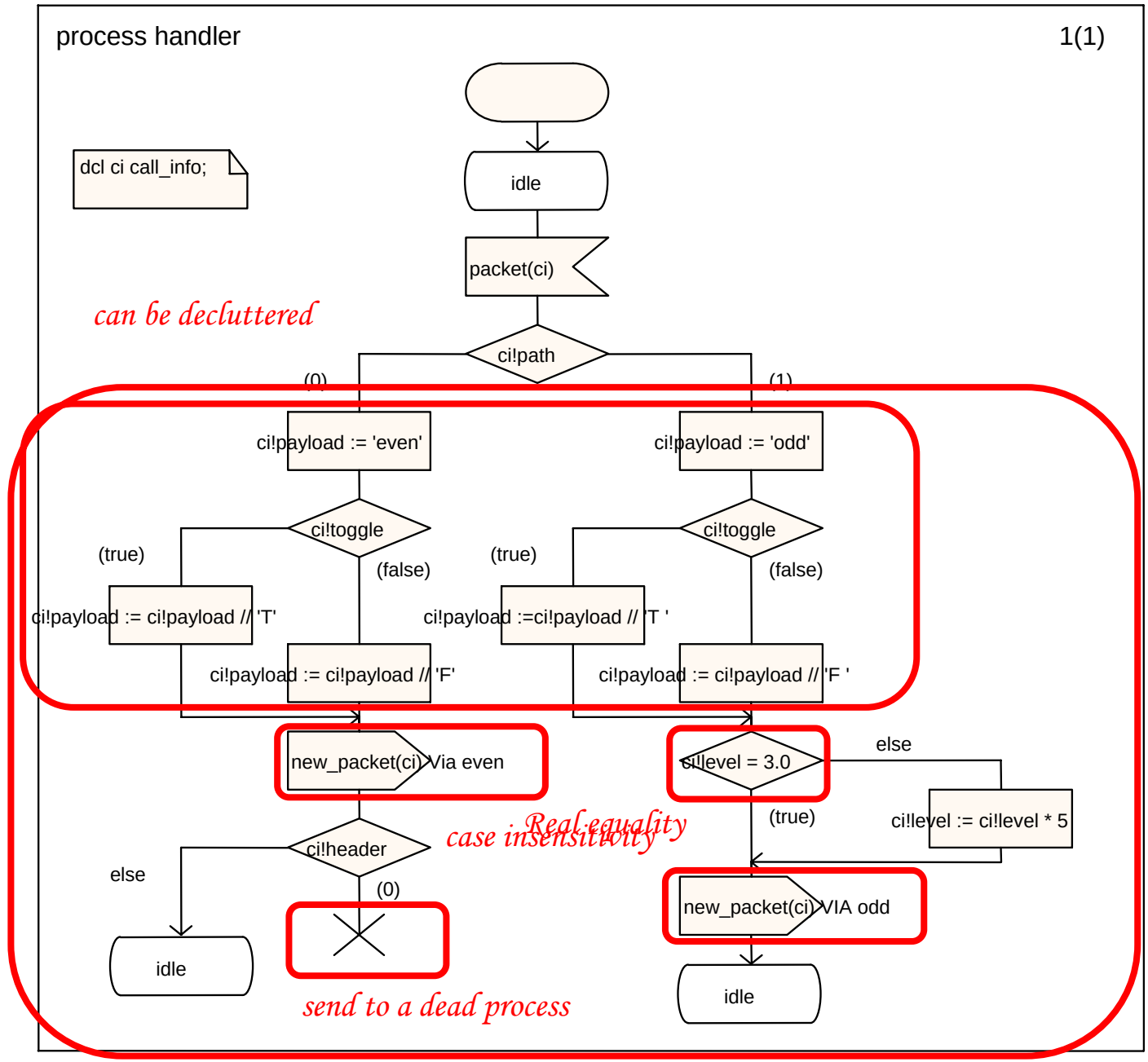
loop control is wrong

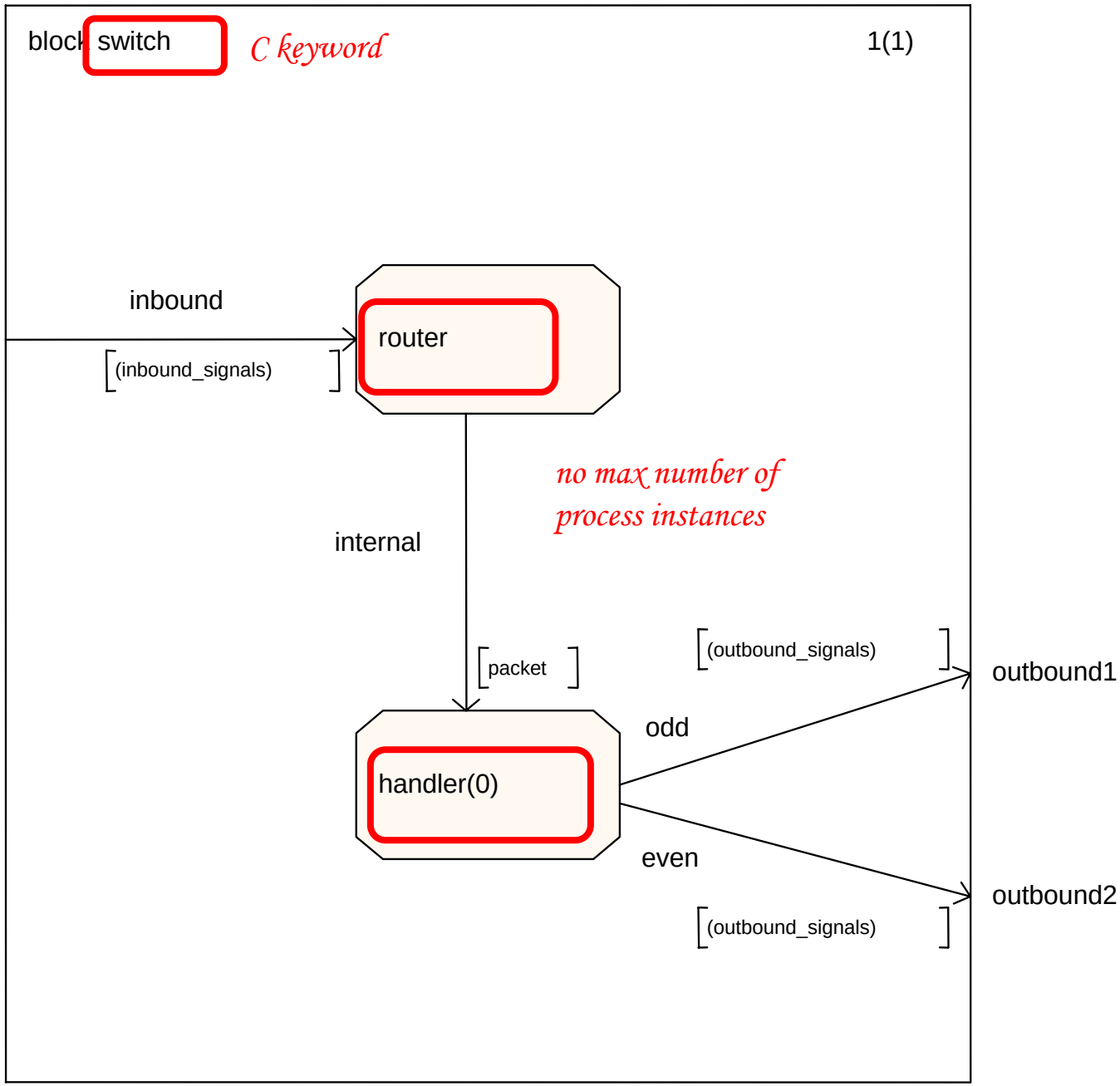
Time literal

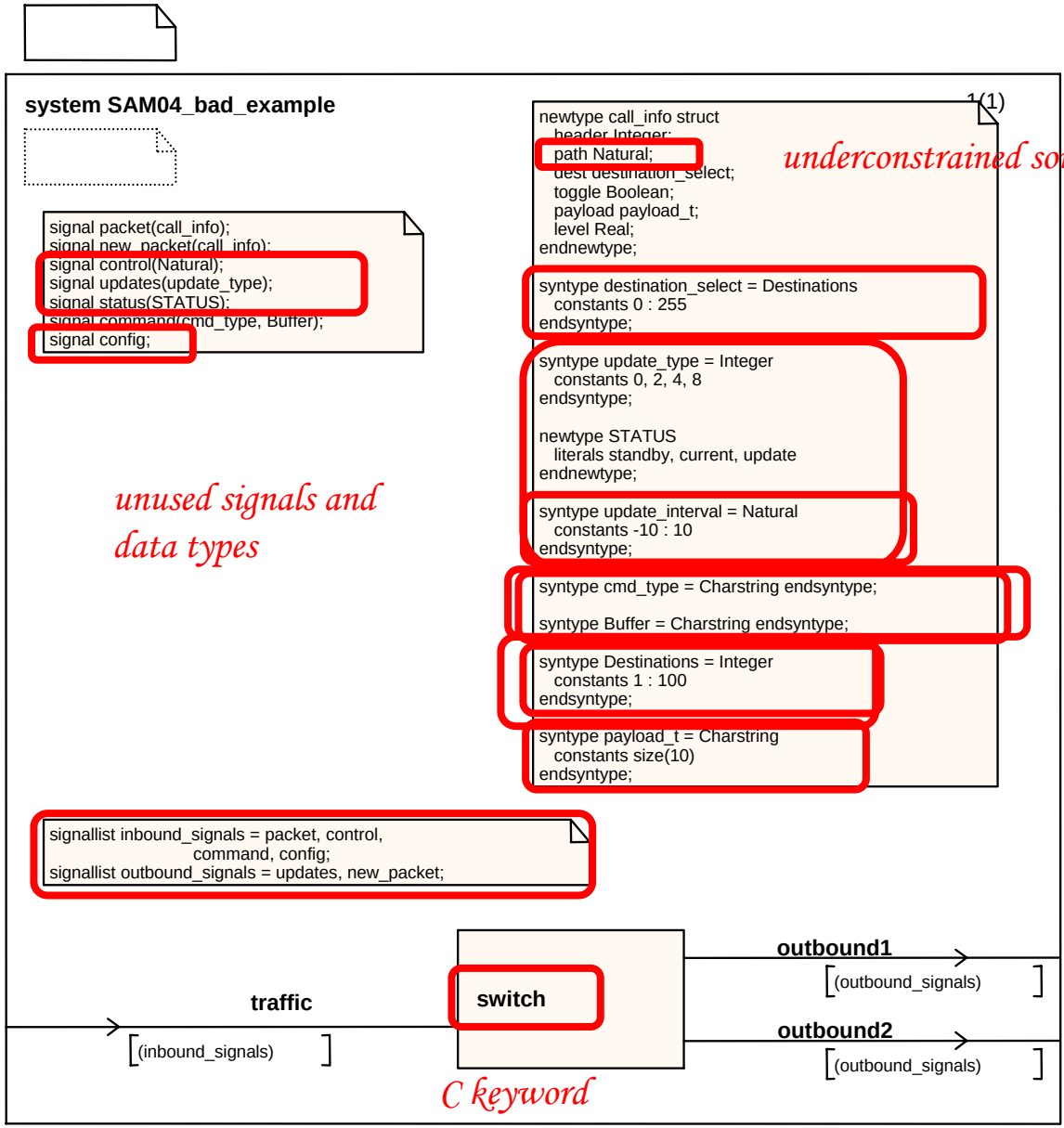
case insensitivity

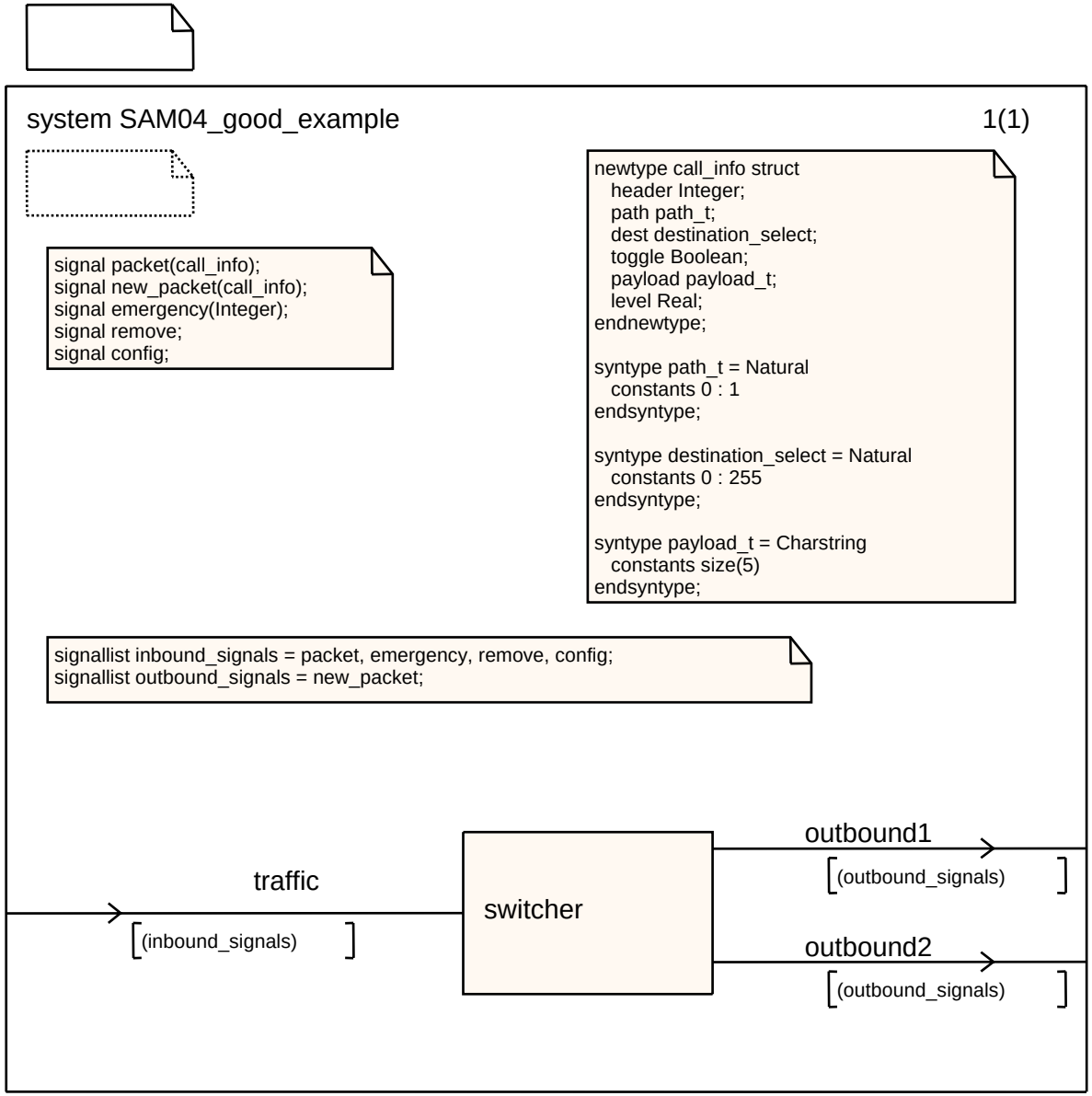
informal text as comment

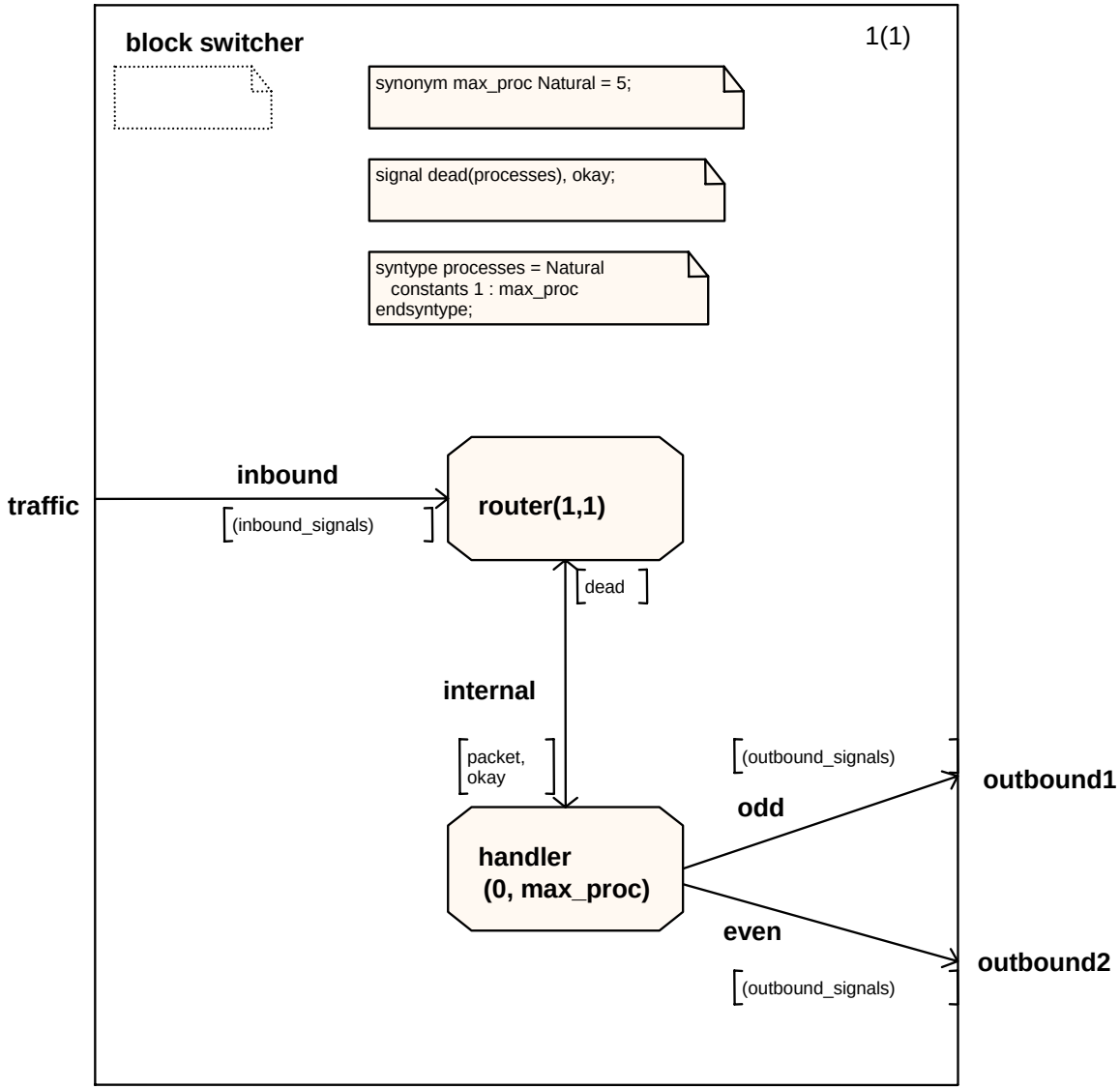

process router
2(2)
procedure log;
fpar in stat Integer;
external;
C reserved word
idle
marshaling
TIM
packet(ci)
command(ct, b)
log(1)
ci_tmp := ci
ct
(REMOVE_CMD)
(911)
invalid String access
Duration precision
set(now+7.325, TIM)
copy of large data structure
d := num(b(0))
ts := not ts
idle
i := 1
idle
possibly incompatible ranges
ci_tmp!dest := d
pow := 2
(false)
ci_tmp!toggle := ts
i <= 15
(true)
ambiguous signal send
packet(ci_tmp) via internal
d := d + num(b(i)) * pow
idle
pow := pow * 2
idle
i := i + 1
can be decluttered


process handler
1(1)
dcl ci call_info;
idle
packet(ci)
can be decluttered
ci!path
(0)
(1)
ci!payload := 'even'
ci!payload := 'odd'
ci!toggle
ci!toggle
(true)
(false)
(true)
(false)
ci!payload := ci!payload // 'T'
ci!payload :=ci!payload // 'T '
ci!payload := ci!payload // 'F'
ci!payload := ci!payload // 'F '
Charstring size constraint violation
new_packet(ci) Via even
ci!level = 3.0
else
(true)
ci!level := ci!level * 5
Real equality case insensitivity
ci!header
else
(0)
new_packet(ci) VIA odd
idle
idle
send to a dead process


block switch
C keyword
1(1)
traffic
inbound
(inbound_signals)
router
no max number of
process instances
internal
packet
(outbound_signals)
outbound1
odd
handler(0)
even
outbound2
(outbound_signals)

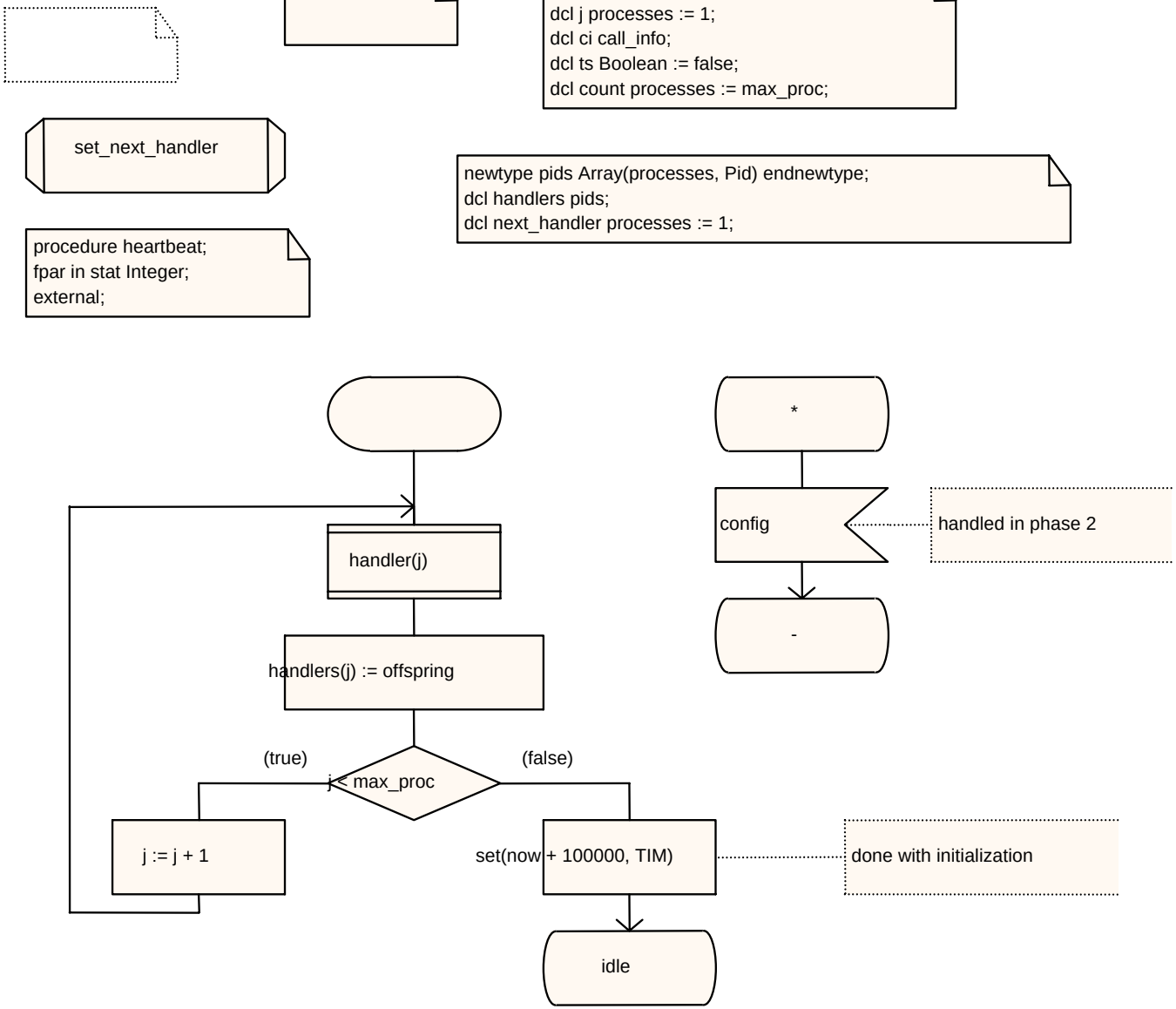

system SAM04_bad_example
1 (1)
signal packet(call_info);
signal new_packet(call_info);
signal control(Natural);
signal updates(update_type);
signal status(STATUS);
signal command(cmd_type, Buffer);
signal config;
newtype call_info struct
header Integer;
path Natural;
dest destination_select;
toggle Boolean;
payload payload_t;
level Real;
endnewtype;
syntype destination_select = Destinations
constants 0 : 255
endsyntype;
syntype update_type = Integer
constants 0, 2, 4, 8
endsyntype;
newtype STATUS
literals standby, current, update
endnewtype;
syntype update_interval = Natural
constants -10 : 10
endsyntype;
syntype cmd_type = Charstring endsyntype;
syntype Buffer = Charstring endsyntype;
syntype Destinations = Integer
constants 1 : 100
endsyntype;
syntype payload_t = Charstring
constants size(10)
endsyntype;
signallist inbound_signals = packet, control,
command, config;
signallist outbound_signals = updates, new_packet;
traffic
(inbound_signals)
switch
outbound1
(outbound_signals)
outbound2
(outbound_signals)
C keyword
underconstrained sort
signal not used in this version
unused signals and data types
nonsensical / confusing data type definitions
marshaling
unbounded / incorrectly bounded aggregate data types


system SAM04_good_example
1(1)
signal packet(call_info);
signal new_packet(call_info);
signal emergency(Integer);
signal remove;
signal config;
newtype call_info struct
header Integer;
path path_t;
dest destination_select;
toggle Boolean;
payload payload_t;
level Real;
endnewtype;
syntype path_t = Natural
constants 0 : 1
endsyntype;
syntype destination_select = Natural
constants 0 : 255
endsyntype;
syntype payload_t = Charstring
constants size(5)
endsyntype;
signallist inbound_signals = packet, emergency, remove, config;
signallist outbound_signals = new_packet;
traffic
(inbound_signals)
switcher
outbound1
(outbound_signals)
outbound2
(outbound_signals)

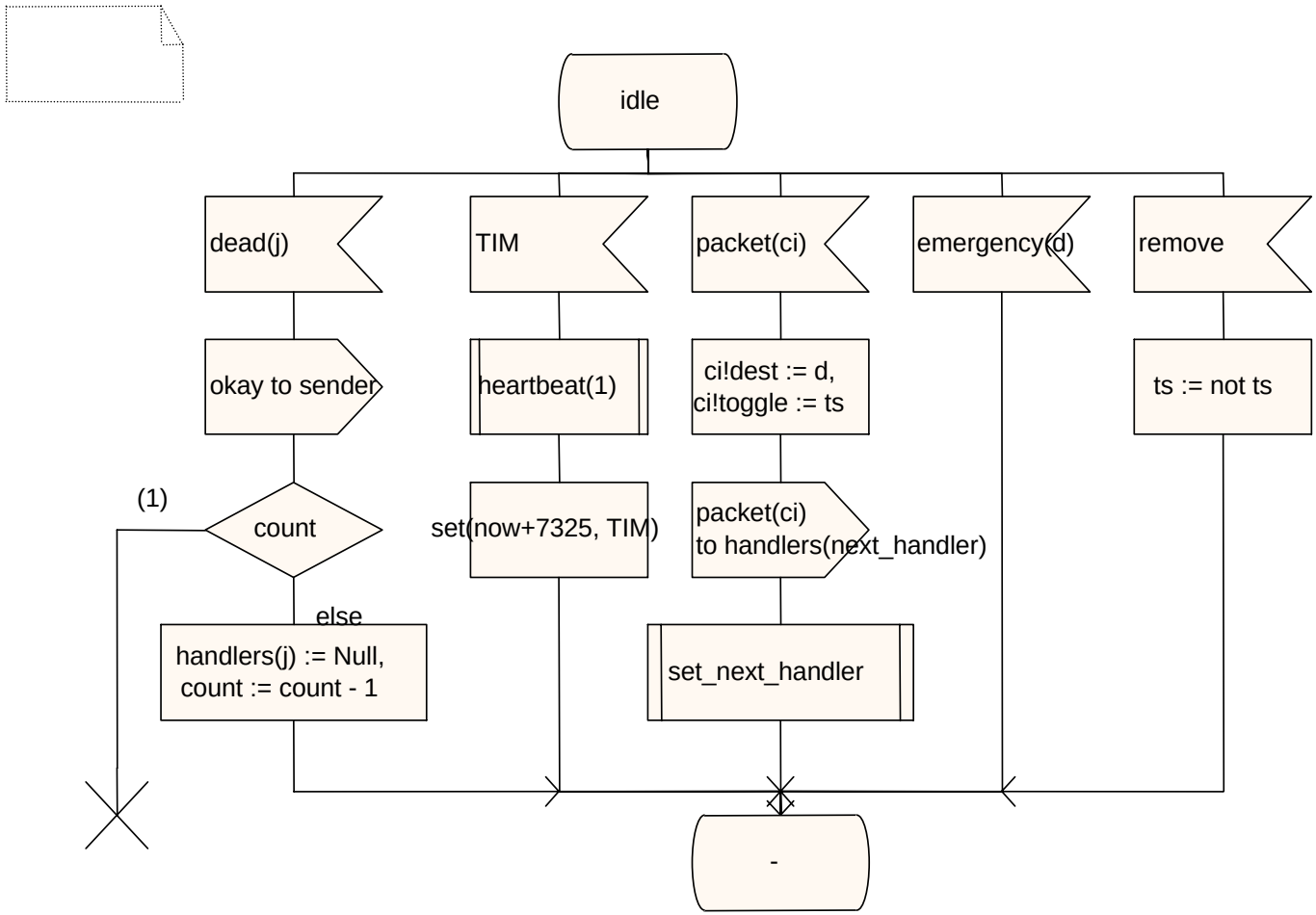

block switcher
1(1)
synonym max_proc Natural = 5;
signal dead(processes), okay;
syntype processes = Natural
constants 1 : max_proc
endsyntype;
traffic
inbound
(inbound_signals)
router(1,1)
dead
internal
packet,
okay
handler
(0, max_proc)
(outbound_signals)
odd
outbound1
even
outbound2
(outbound_signals)

Frank Weil, Frank.Weil@motorola.com

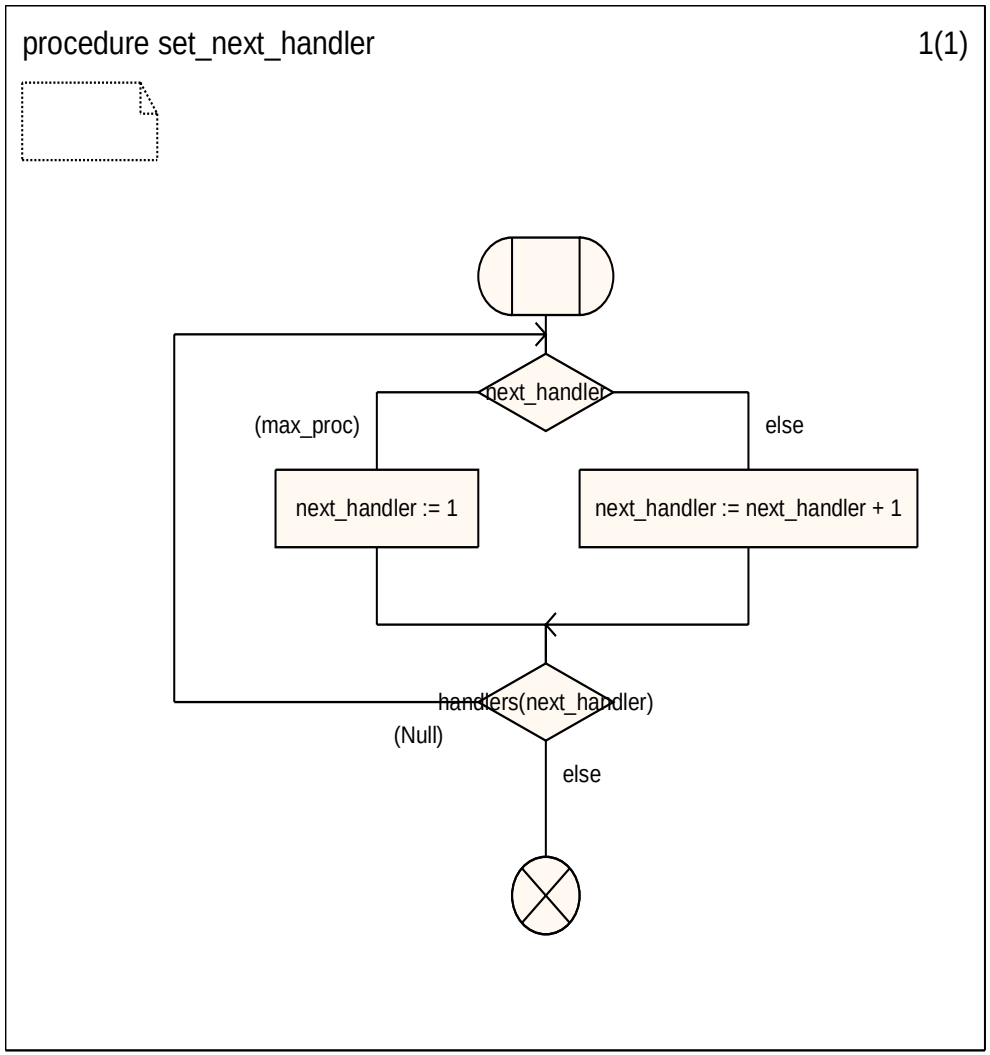
process router



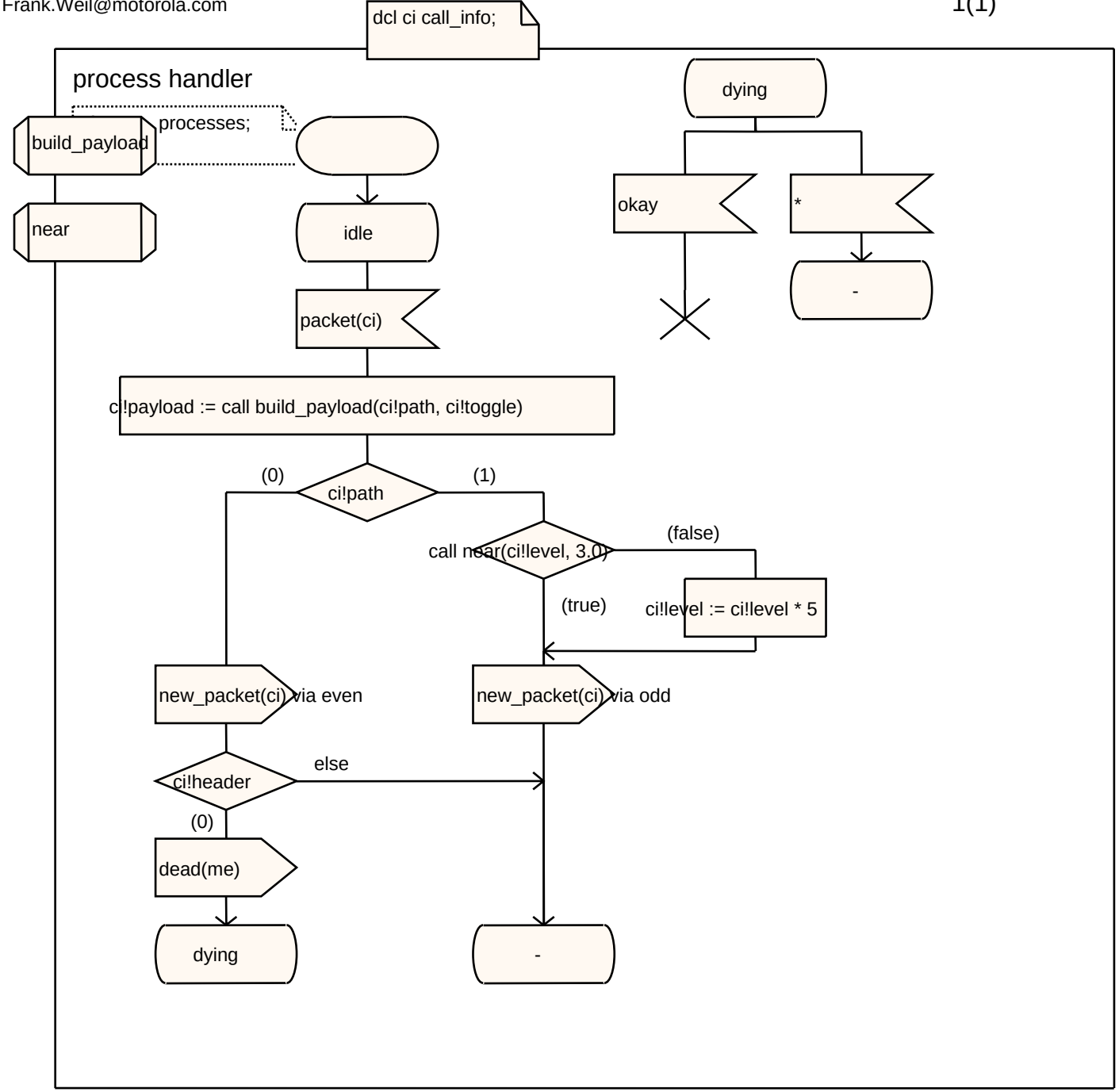
process router **2(2)**

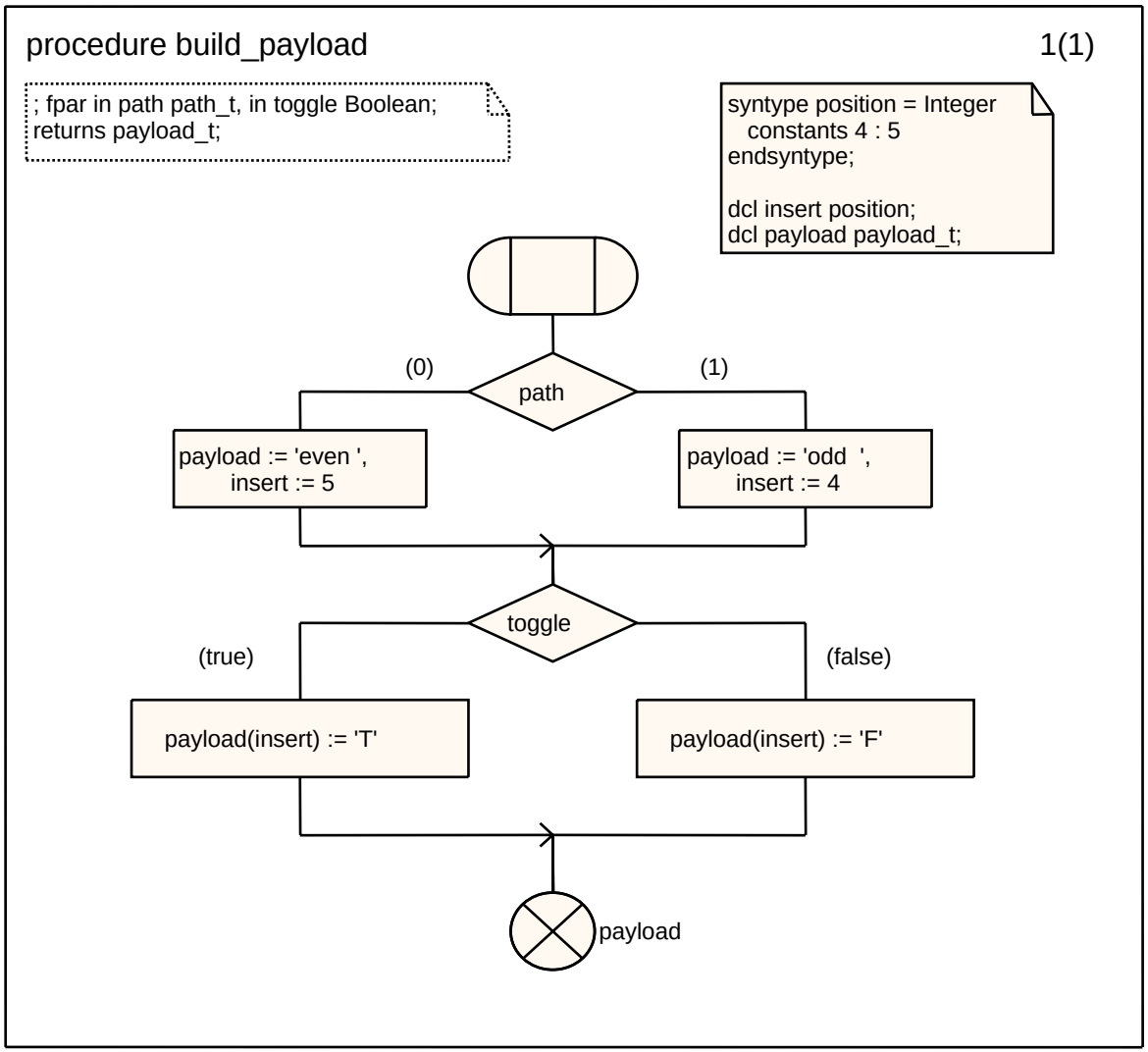


Frank Weil, Frank.Weil@motorola.com



Frank Weil, Frank.Weil@motorola.com




procedure build_payload
1(1)
; fpar in path path_t, in toggle Boolean;
returns payload_t;
syntype position = Integer
constants 4 : 5
endsyntype;
dcl insert position;
dcl payload payload_t;
path
(0)
(1)
payload := 'even ',
insert := 5
payload := 'odd ',
insert := 4
toggle
(true)
(false)
payload(insert) := 'T'
payload(insert) := 'F'
payload

une 2004

procedure near

; fpar in a Real, in b Real;
returns Boolean;

synonym epsilon Real = 0.0001;