Stack Machine

Charles André - UNSA

Advice

- Write the status of the stack as a comment, after each instruction
- Example :

```
  push A        ; | A
  push B        ; | A B
 bcl: cmp       ; stack unchanged and
                ; test of A-B
  beqz done     ; stack unchanged and
                ; branch if A == B
  bltz else     ; stack unchanged and
                ; branch if A < B
  swap          ; | B A
```

gcd

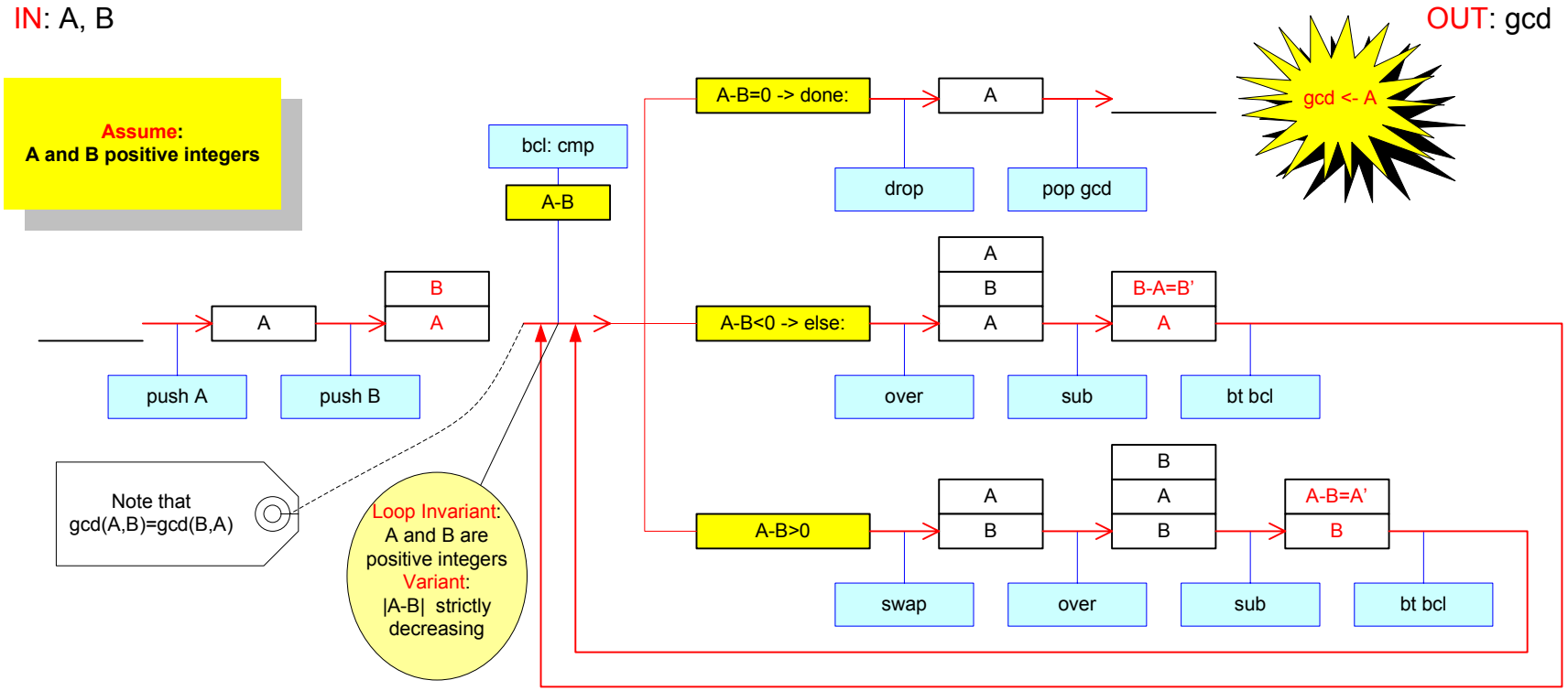
Algorithme: Euclid (by subtraction)

```
// in: A and B: positive integers
// out: gcd(A,B): positive integer
while A ≠ B do
   if A > B then
            A ← A - B
   else
            B ← B - A
   endif
done
gcd ← A
```

Loop Invariant:
A and B positive integers

Variant:
|A-B| strictly decreasing

gcd: stack implementation



Sum of the first N odd numbers

Definition: $$S = \sum_{k=0}^{k=N-1} (2*k+1)$$

Algorithm

```
// in: N: positive integer
// out: S: positive integer
odd:=1:integer;    S = 0; // init
do N times
   S = S + odd;
   odd = odd + 2
done
```

Simple addition
Thanks to the use of a local variable

Advice

- The less memory reference, the better
- Use dup, over to save temporary values in the stack
- Use swap, roll to re-order values in the stack
- Use drop, pop to discard useless values
- Find out orderings that are loop invariants

Stack implementation

```
 ; initializations
 pushi     1              ; |odd
 pushi     0              ; |odd S
 push      N              ; |odd S N
 ; main loop
bcl: pushi      1         ; |odd S N 1
 sub                      ; |odd S N-1=N'
 bltz      done           ; branch if N'<0
 roll                     ; |N' odd S
 over                     ; |N' odd S odd
 add                      ; |N' odd S+odd=S'
 roll                     ; |S' N' odd
```

This structure should be a loop invariant

New value of N

New value of S

Stack implementation

```
 pushi   2              ; |S' N' odd 2
 add                    ; |S' N' odd+2=odd'
 roll                   ; |odd' S' N'
 bt bcl                 ;
; after the loop
done: drop              ; |odd S
 pop     S              ; |odd
                        ; and S memory set
 drop                   ; |
```

New value of odd

Structure Loop invariant