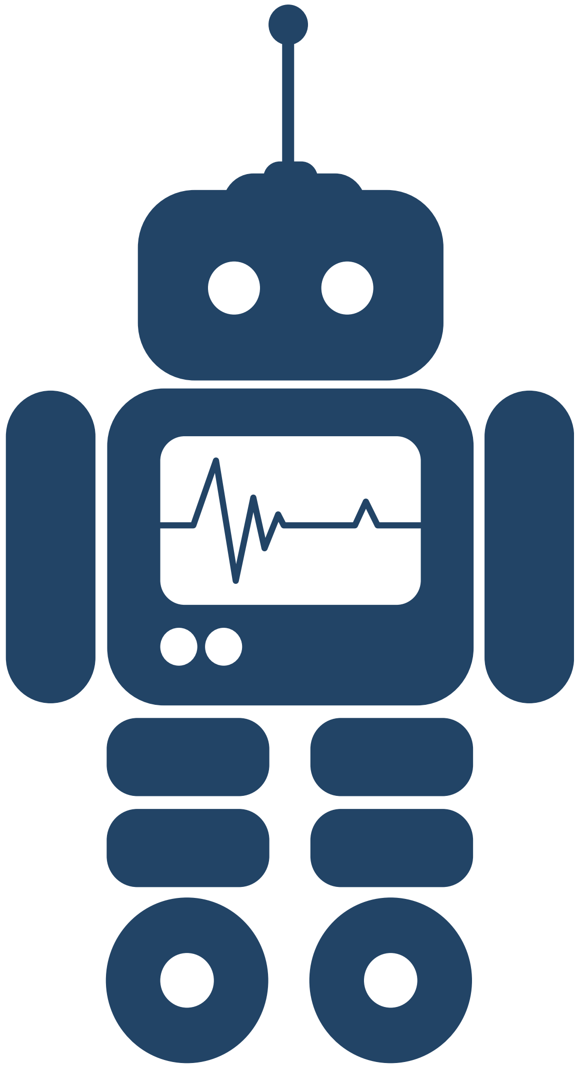
{EPITECH.}
PARIS GRADUATE SCHOOL OF DIGITAL INNOVATION

x86-64 Assembly

Unit & Project Presentation

PARIS - BORDEAUX - LILLE - LYON - MARSEILLE - MONTPELLIER - NANCY - NANTES - NICE - RENNES - STRASBOURG - TOULOUSE



{EPITECH.}
PARIS GRADUATE SCHOOL OF DIGITAL INNOVATION

Unit presentation

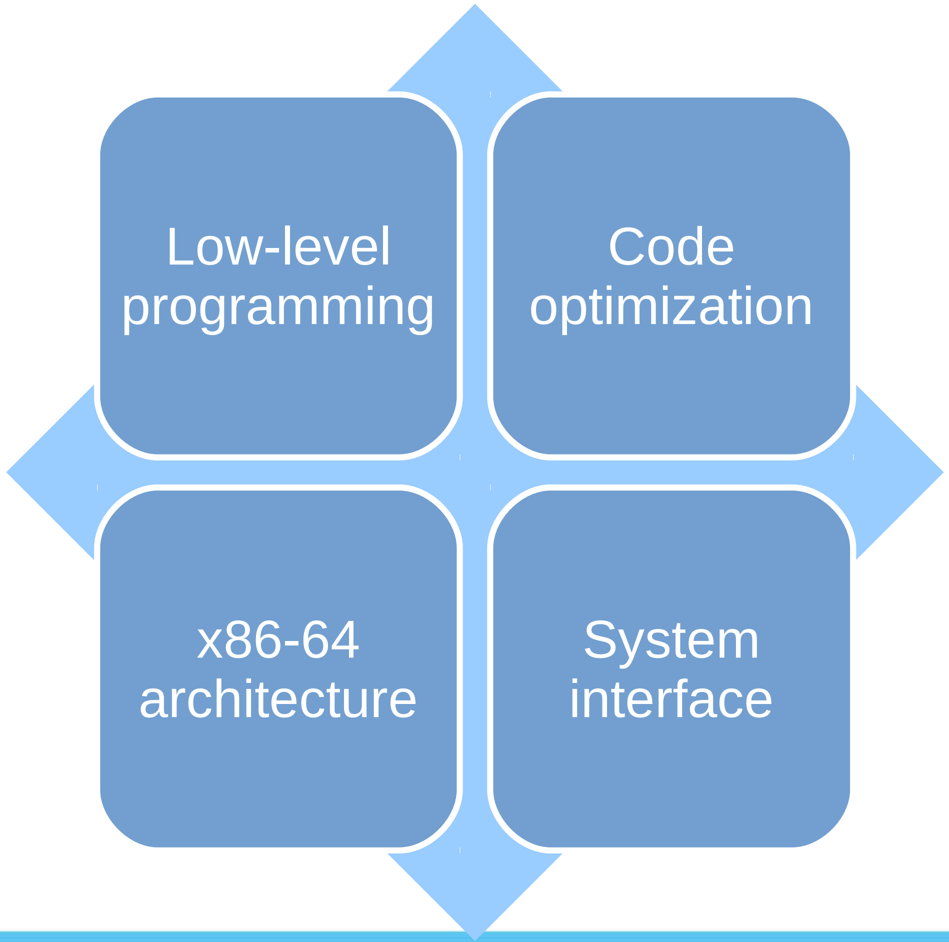
Assembly: What For?

- Better understanding of microprocessor capabilities
- More efficient code writing
- Still used for embedded/kernel programming
- Reverse engineering, debugging

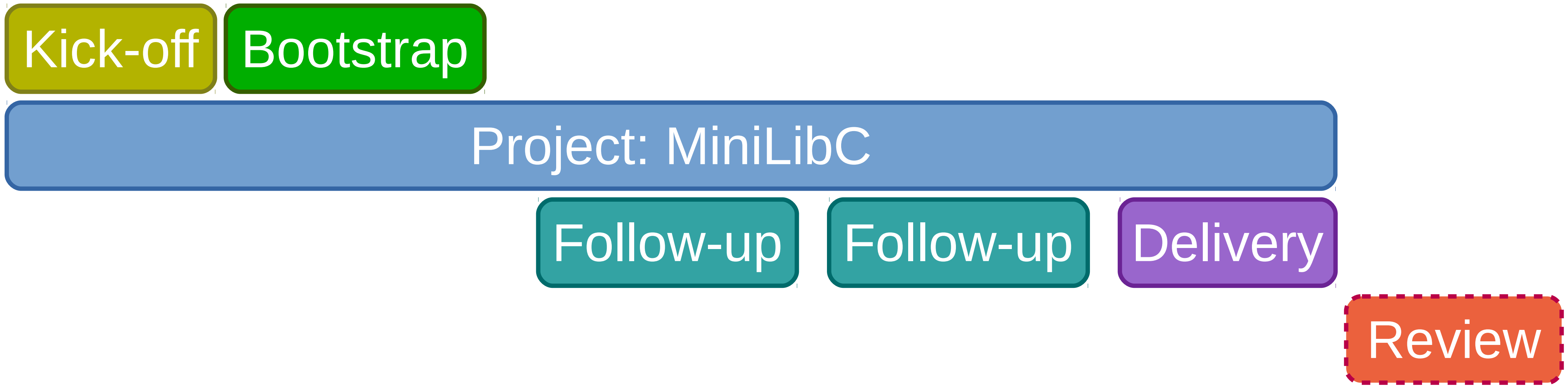
IT Culture

- Turing machine
- Von Neumann architecture
- CISC vs RISC
- Superscalar desings
- Pipelines

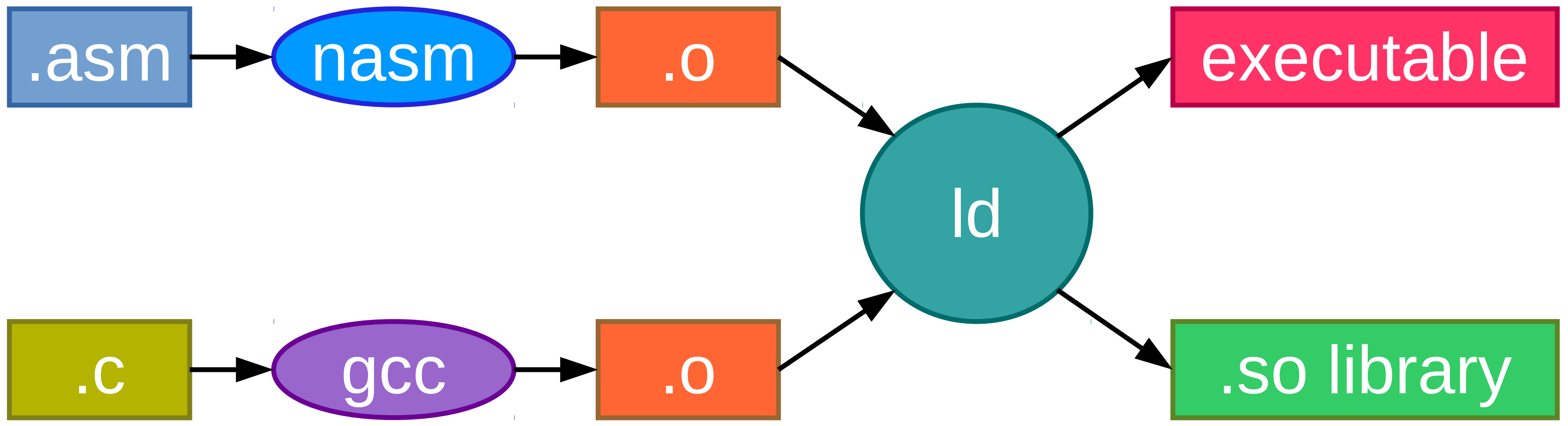
Skills to Be Acquired



Unit Planning



Toolchain



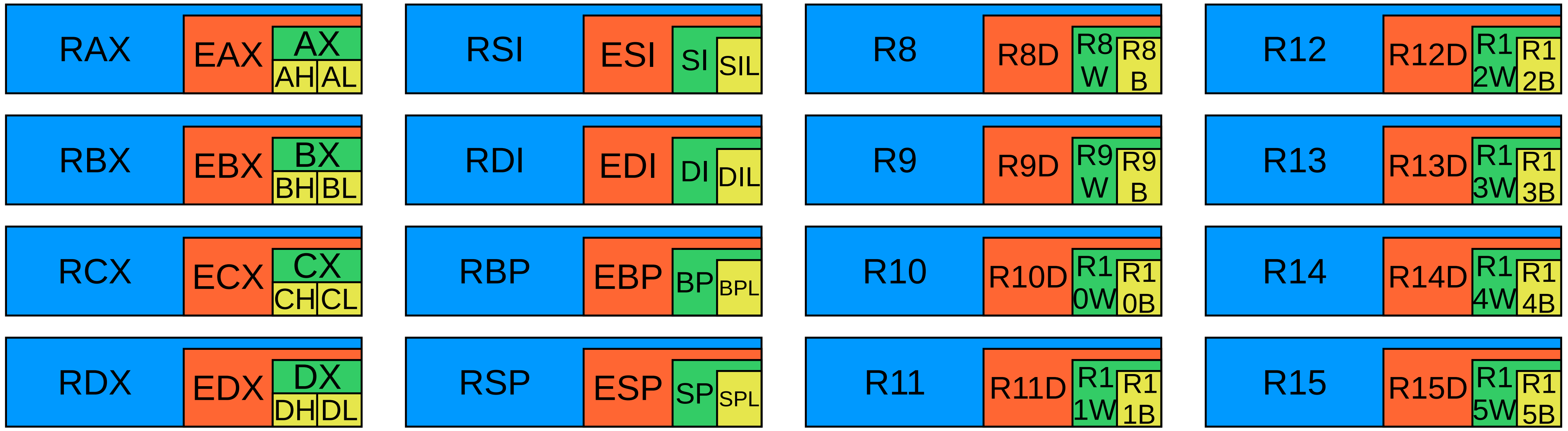
PARIS GRADUATE SCHOOL OF DIGITAL INNOVATION

Registers

General-Purpose Registers (GPR)

- 16 registers to be used as you wish
- However, some have specific roles

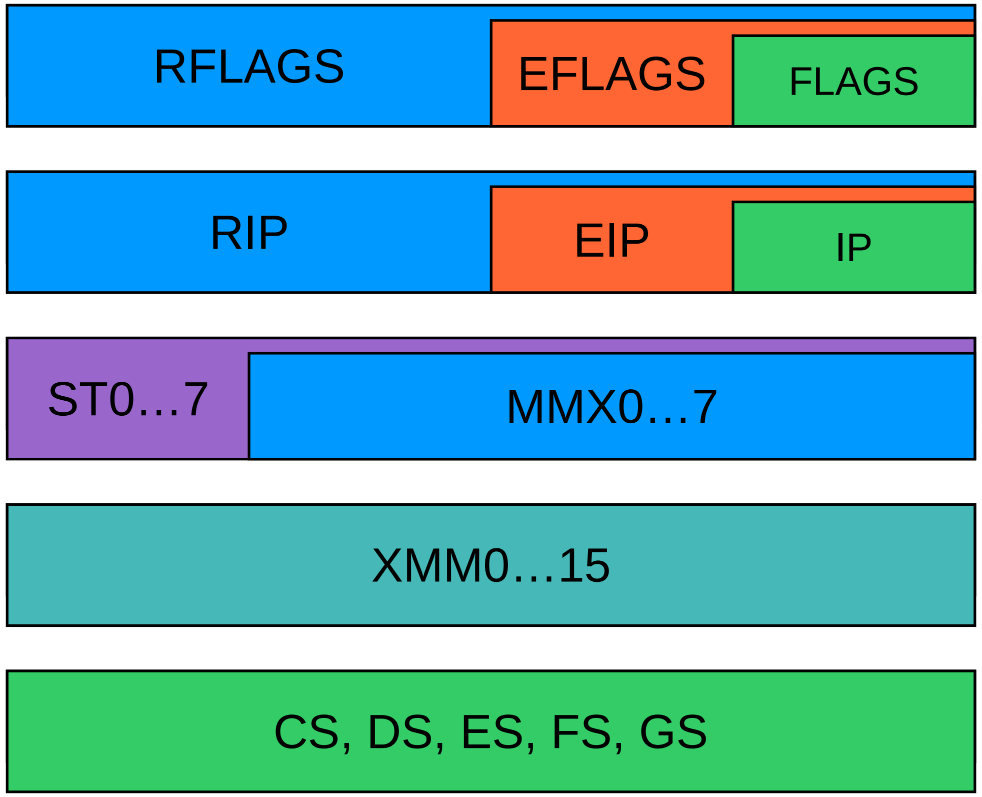
🟦 64-bit 🟩 16-bit
🟥 32-bit 🟨 8-bit



GPR Specific Roles

- RSP: stack pointer (important; used implicitly by PUSH, POP, CALL, RET, ENTER, LEAVE…)
- RBP: frame pointer (optional; used implicitly by ENTER & LEAVE)
- RCX: counter for loop and string instructions
- RSI: source pointer for string instructions
- RDI: destination pointer for string instructions

Other Registers



- RFLAGS: for conditional jumps
- RIP: current instruction pointer
- ST0…7 (80-bit): legacy floating-point numbers
- MMX0…7 (64-bit) & XMM0…15 (128-bit): vector instructions
- Segment registers & numerous other registers: for operating system instructions

Flags

- Set by most instructions; CMP & TEST most appropriate
- Tested by conditional jump instructions J*xx*
- CF: unsigned carry (integer overflow)
- OF: signed overflow
- ZF: zero (result is null)
- SF: sign (result is negative, leftmost bit = 1)
- PF: parity (rightmost bit = 0)

Function Calling Conventions

- Specified by the System V AMD64 Application Binary Interface (ABI)
- 6 first integer/pointer parameters in RDI, RSI, RDX, RCX, R8 & R9
- 8 first floating-point number (FPN) parameters in XMM0…7
- Remaining parameters in the stack
- VarArgs (printf…): number of FPN parameters in RAX
- RBP, RBX, R12, R13, R14 & R15: must be preserved by callee
- Other registers may be altered at will
- Return value in RAX (integer/pointer) or XMM0 (FPN)

System Call

- Same registers used, except R10 instead of RCX for 4th parameter
- Integer and pointers only, no floating-point parameter
- System call number in RAX
- RCX & R11 may be overwritten
- Specific instruction: SYSCALL
- Return value in RAX, on error RAX = −*errno* (between −4095 and −1)
- List: /usr/include/asm/unistd_64.h

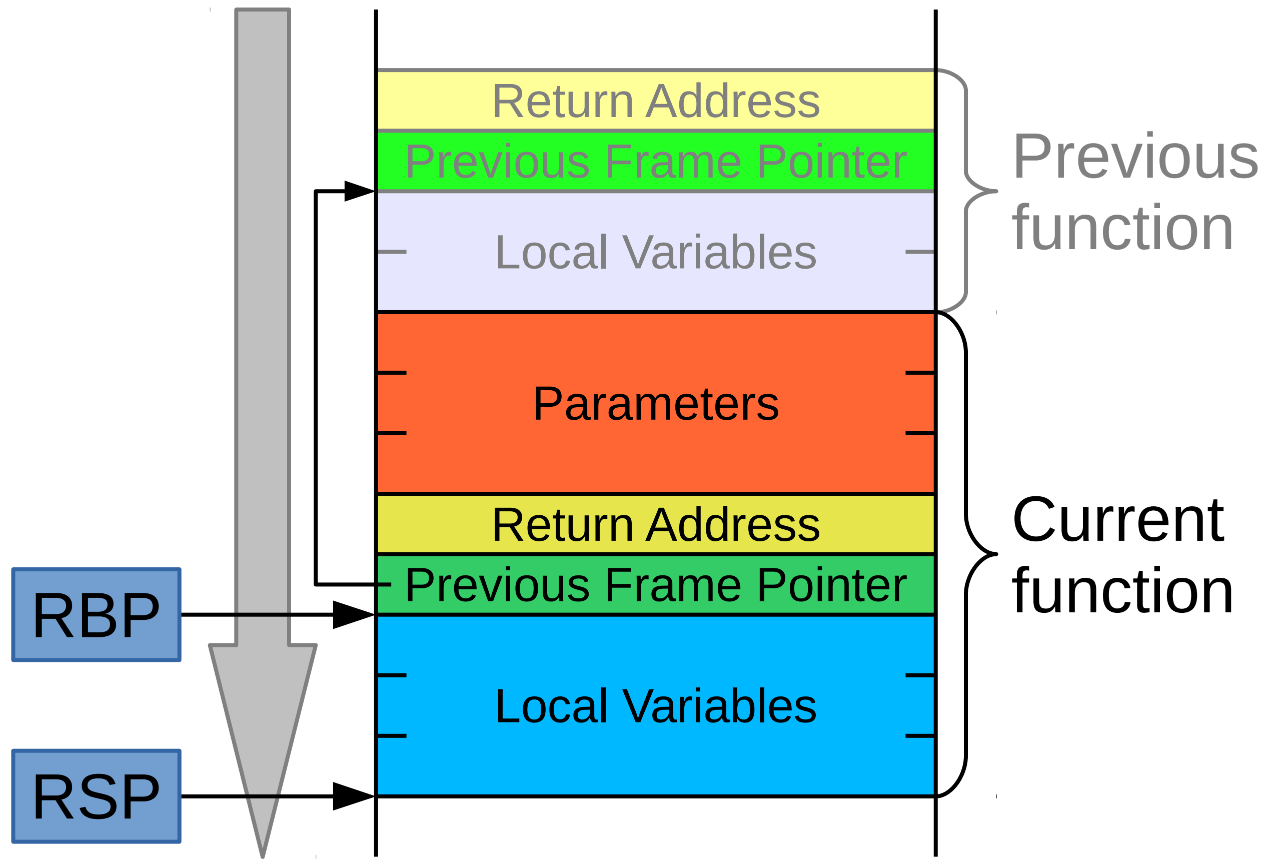
PARIS GRADUATE SCHOOL OF DIGITAL INNOVATION

Memory

Static Sections

- Code: .text
- Read-only data: .rodata
- Read/write data: .data
- Unitialized data: .bss
- Cf. nm/objdump

Stack



- Grows downwards
- Contains the function local state
- RSP = top of stack
- RBP = frame pointer (beginning of local variables), for convenience only

Stack Frame Setup

- Optional
- RSP varies (pushes and pops), but RBP fixed
- Easier to access local variables: *RBP – constant*
- Function prologue: `PUSH RBP + MOV RBP, RSP`
- Function epilogue: `MOV RSP, RBP + POP RBP`
- Alternatives: ENTER & LEAVE

{EPITECH.}
PARIS GRADUATE SCHOOL OF DIGITAL INNOVATION

Instructions

Assembly Directives

- Comment: `; this is a comment`
- Set architecture: `BITS 64`
- Change section: `SECTION` *`.name`*
- Set symbol: *`label`*`:`
- Export symbol: `GLOBAL` *`symbol`*
- Import symbol: `EXTERN` *`symbol`*
- Put bytes (static data): `DB <`*`data`*`>` / `RESB <`*`length`*`>`

Instruction Syntax

- *`INSTR SRC/DEST`* or *`INSTR SRC/DEST, SRC`*
- Source (*SRC*): immediate value, register or memory
- Destination (*DEST*): register or memory
- Both arguments cannot be memory
- Type determined from other parameter if available or specified explicitly using a prefix (BYTE, WORD, DWORD, QWORD)
- Example: ADD RAX, RDX ⇔ RAX = RAX + RDX

Memory Access

- `[`*`immediate`*` + `*`register`*` + `*`register`*` * `*`coefficient`*`]`
  - *immediate*: immediate value (explicit constant)
  - *register*: general-purpose register
  - *coefficient*: 1 (default), 2, 4 or 8
  - All are optional
- Type is determined from other parameter if available or can be specified explicitly using a prefix (BYTE, WORD, DWORD, QWORD)
- Example 1: `MOV RDX, [RBX + RCX * 4]`
- Example 2: `MOV BYTE [RDI + 1337], 42`

Main Instructions

- Data movement: MOV, XCHG, PUSH, POP
- Type conversion: CBW, CWDE, CDQE
- Arithmetic: NEG, INC, DEC, ADD, SUB, IMUL, MUL, IDIV, DIV
- Bitwise: NOT, AND, OR, XOR
- Bitshifts: SHL, SHR, SAL, SAR, ROL, ROR
- Resultless (flags only, for conditional jumps): CMP, TEST

Branching

- Unconditional jump: JMP
- Conditional jump (depends on RFLAGS): JA, JAE, JB, JBE, JC, JE, JG, JGE, JL, JLE, JNA, JNAE, JNB, JNBE, JNC, JNE, JNG, JNGE, JNL, JNLE, JNO, JNP, JNS, JNZ, JO, JP, JPE, JPO, JS, JZ (cheers!)
- Function call: CALL (‘PUSH RIP’ + JMP)
- Function return: RET (‘POP RIP’)
- System function call (kernel interface): SYSCALL

Miscellaneous

- No-operation: NOP (actually: XCHG RAX, RAX), used to fill up space
- Instructions with carry
- String instructions (with REP prefix)
- LEA: immediate + register + register × coefficient, all at once
- Supplemental instruction sets: MMX, SSE, AVX, AES-NI…
- …+ many rarely-used, legacy and system instructions

Ressources

- Intel Architectures Software Developer Manuals, Volumes 1 & 2
- NASM documentation
- The Internet

EPITECH.

PARIS GRADUATE SCHOOL OF DIGITAL INNOVATION

Example

```
BITS 64                                 ; 64-bit mode

SECTION .text                           ; Code section

GLOBAL  main                            ; Export 'main'
EXTERN  printf                          ; Import 'printf'

main:
        PUSH    RBP                     ; Prologue:
        MOV     RBP, RSP                ; Stack frame setup

        MOV     RDI, str                ; First parameter
        CALL    printf                  ; Function call: printf(str)
        ;; WARNING: won't work! (cf. calling conventions)

        MOV     RAX, 60                 ; exit() syscall number
        XOR     RDI, RDI                ; RDI = 0 (first parameter)
        SYSCALL                         ; System call: exit(0)

        LEAVE                           ; Epilogue
        RET                             ; Return

SECTION .rodata                         ; Read-only data

str:    DB 'Hello, World!', 0Ah, 0      ; Format string for printf()
```

{EPITECH.}
PARIS GRADUATE SCHOOL OF DIGITAL INNOVATION

Questions?