絕對載入器開發報告

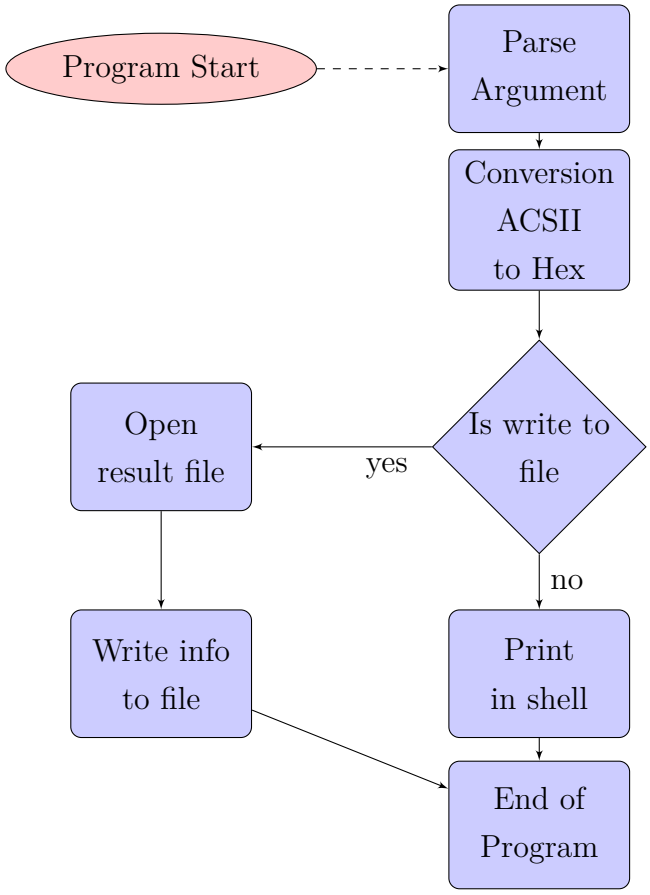
姓名：黃毓峰　　學號：S11159005

2024/06/10

1　摘要

根據課本的 Absolute Loader 的設計構想，並利用 C 來實做出來，該設計出的程式可以讀取以由 ACSII Code 編碼組成的一串 16 進位數值，並根據給予的記憶體起始位置輸出載入後的記憶體位置，與其改位置的記記憶體數值，可以選在在 Shell or Console 中顯示或是把結果寫入在文件。

2　流程圖



3 設計方法

3.1 Parse Argument

定義了一 enumeration 來標示該程式共有記個 argument 可以做使用

```
enum ARG_OPTS {
  START_ADDRESS,
  IS_OUT_RESULT,
};
```

並定義了一 function 來把 enumeration 轉成對應的字串

```
int argtoc(enum ARG_OPTS options, char **result) {
if (result == NULL || *result == NULL) {
  fprintf(
    stderr,
    "argument to string failed : null pointer\n"
  );
  return -1;
}

switch (options) {

  case START_ADDRESS:
  strcpy(*result, "-A");
  return 0;
  return -1;
}
}
```

用來存解析結果

```
struct args {
  char* program_path;
  char* start_address;
  char* file_path;
  bool is_out_result;
};
```

用來則解析程式的 argument 的部份程式

```
argtoc(START_ADDRESS, &opt_str);
opt_len = strlen(opt_str);

if (strncmp(arg, opt_str, opt_len) == 0) {

  if (arg_len == opt_len) {
    // move to next argument to get the file path
    arg_index++;

    if (get_arg(argc, argv, arg_index, &arg, &arg_len) < 0) {
      exit(-1);
    }

    result->start_address = arg;
    continue;
  }

  result->start_address = (arg + opt_len);
  continue;
}
```

3.2 Conversion ACSII to Hex

3.2.1 Number from ASCII

因不給使用 atoi, 以此猜測也不開放使用 strtol 等等得的 function, 故重新設計一把 ASCII 的字串轉成特定 base 的 function，以下為真對 base 為 16 的轉換程式，這樣就可以把 Memory Start Address (argument 給的) 轉成數字，也可以把輸入文件的 ACSII 轉成對應的半位元

```
int num_from_ascii(
  const char* accii, const uint32_t ascii_len, const int base
) {
  int result = 0;
  switch (base) {
    case 0x10:
      for (uint32_t index = 0; index < ascii_len; index++) {

        // get the character of string
        char c = accii[index];

        // get dec num from ascii
        c -= '0';

        // more the pure number (9)
        c = (c > (10 - 1)) ? c - (0x10 - (10 - 1)) : c;

        // if character is upper case, sub offset
        c = (c > (0x10 - 1)) ? c - ('a' - 'A') : c;

        // carrying
        result *= base;

        // add number
        result += c;
      }
      break;
    default:
      fprintf(stderr, "num_from_ascii : unknow base");
      return 0;
  }
  return result;
}
```

3.2.2 合併半位元

為了方便做合併動作定一結構為 hex_wf (hex with half)

```
struct hex_wf {
  uint8_t rf : 4;
  uint8_t lf : 4;
};
```

這樣可以方便且快數（不用做位移以及 or bitwise) 可以取得合併 2 個半位元

```
struct hex_wf* data = (struct hex_wf *)malloc(sizeof(struct hex_wf));
data->lf = 0xf;
data->rf = 0x1;
uint8_t fullbyte = *((uint8_t*) data);
```

把合併後的結果存在 list 裡面並記入 list 長度, 待全部完成後把 list 轉成 array

```
*dest = (uint8_t*) malloc(sizeof(uint8_t) * (*dest_len));

// to for each element of list
struct list_head *pos, *n;

uint32_t index = 0;
list_for_each_safe(pos, n, &hex_wf_list)
{
  struct list_node_hex_wf *st = list_entry(
    pos,struct list_node_hex_wf, list
  );
  uint8_t content = *((uint8_t*) st->data);
  (*dest)[index] = content;
  index++;
  list_del(pos);
  free(st);
}
```

詳細程式碼請見 project_root/lib/core/loader.c 的 conversion_hex

3.3 Wirte info to File & Print in Shell

藉由上述的步驟可以取的記憶體起始位置，輸入文件載入後的 16 進位數值，最後要做輸出的部份為了能夠輸出到不同的 IO 流又不用寫太多重複的程式碼於是設計了一 function 可以把資訊寫入到 byffer （char array）理面, 最後在輸出到不同的 IO 流

```
1  // storage loading file
2  uint8_t *hex_array  = NULL;
3  uint32_t hex_len = 0;
4
5  // conversion ascii to hex (byte)
6  conversion_hex(arg_result.file_path, &hex_array,  &hex_len);
7
8  // init memory info str buffer
9  char *buffer = (char *) malloc(sizeof(char) * CONSOLE_MESSAGE_BUFFER_LEN
10 memset(buffer, '\0', CONSOLE_MESSAGE_BUFFER_LEN);
11
12 // show memory data or wirte to output file
13 for (int i = 0; i < hex_len; i++) {
14
15   // print info to buffer
16   sprint_memory_info(&buffer, address, 5, hex_array[i]);
17
18   // add memory address
19   address += sizeof(uint8_t);
20   // select IO (stdout, or outputfile)
21   if (arg_result.is_out_result) {
22     fprintf(out_file, "%s", buffer);
23   }
24   else {
25     fprintf(stdout, "%s", buffer);
26   }
27 }
```

4 開發環境

作業系統	**Linux 6.6.32-1-lts x86_64**
桌面環境	**KDE Plasma 6.0.5, Qt 6.7.1 , Wayland 1.23.0-1**
CMake	**3.29.5**
GCC	**14.1.1**
Make	**4.4.1**

5 討論與心得

5.1 討論

讀文件減數值輸出

5.2 心得

最近好累好忙，下次忙的時候不用 latex 寫學校報告了