Liste des sections traitées

- 0x00000000fffffff0:0x00000000fffffff4 CODE16
- 0x00000000ffffff40:0x00000000ffffff6a CODE16
- 0x00000000ffffffb8:0x00000000ffffffbd GDTR32
- 0x00000000ffffff70:0x00000000ffffffb7 GDT32
- 0x00000000fffffe66:0x00000000fffffe80 CODE32
- 0x00000000fffffe81:0x00000000fffffe93 CODE32
- 0x00000000fffffe94:0x00000000fffffe9a CODE32
- 0x00000000fffffe62:0x00000000fffffe65 DATA32
- 0x00000000fffffe9b:0x00000000fffffea4 CODE32
- 0x00000000fffffea5:0x00000000ffffff12 CODE32
- 0x00000000fffffd8d:0x00000000fffffe61 CODE32

Passage en mode protégé

```
d.bios.rom:     format de fichier binary


Déassemblage de la section .data:

fffffff0 <.data+0x3ffff0>:
fffffff0:   0f 09                   wbinvd
fffffff2:   e9 4b ff                jmp    0xffffff40
```

```
d.bios.rom:     format de fichier binary


Déassemblage de la section .data:

ffffff40 <.data+0x3fff40>:
ffffff40:   db e3                   fninit
ffffff42:   0f 6e c0                movd   %eax,%mm0
ffffff45:   fa                      cli
ffffff46:   66 2e 0f 01 16 b8 ff    lgdtl  %cs:-0x48
ffffff4d:   0f 20 c0                mov    %cr0,%eax
ffffff50:   0c 01                   or     $0x1,%al
ffffff52:   0f 22 c0                mov    %eax,%cr0
ffffff55:   fc                      cld
ffffff56:   b8 08 00                mov    $0x8,%ax
ffffff59:   8e d8                   mov    %ax,%ds
ffffff5b:   8e c0                   mov    %ax,%es
ffffff5d:   8e d0                   mov    %ax,%ss
ffffff5f:   8e e0                   mov    %ax,%fs
ffffff61:   8e e8                   mov    %ax,%gs
ffffff63:   66 ea 66 fe ff ff 10    ljmpl  $0x10,$0xfffffe66
ffffff6a:   00
```

```
00000000ffffffb8 0047-ffffff70
```

```
00000000ffffff70 00000000 00000000
00000000ffffff78 0000ffff 00cf9300
00000000ffffff80 0000ffff 00cf9b00
00000000ffffff88 0000ffff 00cf9300
00000000ffffff90 0000ffff 00cf9b00
00000000ffffff98 00000000 00000000
00000000ffffffa0 0000ffff 00cf9300
00000000ffffffa8 0000ffff 00af9b00
00000000ffffffb0 00000000 00000000
```

Le bios commence par sauter à une routine permettant de passer en mode protégé. La GDT est en mode FLAT pour chacun de ses descripteurs de segments. A présent, le processeur est en mode protégé.

Fonction `0xfffffe66`

```
d.bios.rom:     format de fichier binary


Déassemblage de la section .data:

fffffe66 <.data+0x3ffe66>:
fffffe66:   b8 60 00 00 80          mov    $0x80000060,%eax
fffffe6b:   66 ba f8 0c             mov    $0xcf8,%dx
fffffe6f:   ef                      out    %eax,(%dx)
fffffe70:   66 ba fc 0c             mov    $0xcfc,%dx
fffffe74:   b8 04 00 00 00          mov    $0x4,%eax
fffffe79:   ef                      out    %eax,(%dx)
fffffe7a:   ed                      in     (%dx),%eax
fffffe7b:   0d 01 00 00 f8          or     $0xf8000001,%eax
```

Le controlleur mémoire du processeur est initialisé. Il est configurable à travers l'espace PCI en utilisant les I/O. Son identifiant est : `B0:D0:F0`. Pour obtenir ces informations, il suffit d'appliquer les formules suivantes :

```
address = 0x80000060
B = (address - 0x80000000) >> 16
D = ((address - 0x80000000) >> 11) & 31
F = ((address - 0x80000000) >>  8) & 7
register = (address - 0x80000000) & 255
```

Dans son espace de configuration, le registre `0x60` correspond au `PCIEXBAR`. L'espace MMIO est donc configuré pour être adressé en `0xf8000000`. Le `4` signifie que seulement `64Mo` seront adressables, ce qui est cohérent avec les composants dans le portable et ce qui évite les gaspillage de mémoire. Le pseudo-code est le suivant :

```
io(0xcf8) = 0x80000060
io(0xcfc) = 0x4
io(0xcfc) = io(0xcfc) | 0xf8000001
```

Ces informantions sont disponibles dans le document :

Mobile 3rd Generation Intel CoreTM Processor Family, Mobile Intel Pentium Processor Family, and Mobile Intel Celeron Processor Family

---

```
d.bios.rom:     format de fichier binary


Déassemblage de la section .data:

fffffe81 <.data+0x3ffe81>:
fffffe81:   bf f0 80 0f f8          mov    $0xf80f80f0,%edi
fffffe86:   c7 07 01 c0 d1 fe       movl   $0xfed1c001,(%edi)
fffffe8c:   bf 10 f4 d1 fe          mov    $0xfed1f410,%edi
fffffe91:   83 27 fb                andl   $0xfffffffb,(%edi)
```

L'espace `0xf80f80f0` correspond au composant `B0:D0:F0`. Pour obtenir cette information, il suffit d'appliquer les formules suivantes :

```
address = 0xf80f80f0
PCIEXBAR = 0xf8000000
R = address % 4096
F = ((address - PCIEXBAR) / 4096) % 8
D = ((address - PCIEXBAR) / (4096 * 8)) % 32
B = ((address - PCIEXBAR) / (4096 * 8 * 32))
```

Ce composant correspond à l'interface avec le bus LPC (*LPC Interface Bridge Registers*). Le registre `0xf0` est l'adresse du RCBA (*Root Complex Base Address*). Cette zone contient les registres de configuration du chipset. Elle est à présent accessible à l'adresse `0xfed1c000`. L'offset `0xfed1f410 - 0xfed1c000` correspond au registre `0x3410` de cette zone, le GCS (*General Control and Status*). En masquant ce registre avec `0xfffffffb`, le bios positionne à `0` le bit RPR : **TODO : mieux comprendre ce bit**

> Reserved Page Route (RPR) — R/W. Determines where to send the reserved page registers. These addresses are sent to PCI or LPC for the purpose of generating POST codes. The I/O addresses modified by this field are: 80h, 84h, 85h, 86h, 88h, 8Ch, 8Dh, and 8Eh.
>
> 0 = Writes will be forwarded to LPC, shadowed within the PCH, and reads will be returned from the internal shadow
>
> 1 = Writes will be forwarded to PCI, shadowed within the PCH, and reads will be returned from the internal shadow.
>
> NOTE: if some writes are done to LPC/PCI to these I/O ranges, and then this bit is flipped, such that writes will now go to the other interface, the reads will not return what was last written. Shadowing is performed on each interface. The aliases for these registers, at 90h, 94h, 95h, 96h, 98h, 9Ch, 9Dh, and 9Eh, are always decoded to LPC.

Ces informantions sont disponibles dans le document :

> Intel 6 Series Chipset and Intel C200 Series Chipset

```
d.bios.rom:     format de fichier binary


```

```
Déassemblage de la section .data:

fffffe94 <.data+0x3ffe94>:
fffffe94:   66 b8 01 00             mov    $0x1,%ax
fffffe98:   66 e7 80                out    %ax,$0x80
```

Le port `0x80` semble être utilisé comment port de diagnostic :

> I/O port 0x80 is traditionally used for POST Codes. (POST = Power On Self Test)

```
00000000fffffe62 fffffea5
```

```
d.bios.rom:     format de fichier binary


Déassemblage de la section .data:

fffffe9b <.data+0x3ffe9b>:
fffffe9b:   bc 62 fe ff ff          mov    $0xfffffe62,%esp
fffffea0:   e9 e8 fe ff ff          jmp    0xfffffd8d
```

Les instructions aux adresses `0xfffffe9b` et `0xfffffea0` correspondent à un `call`. Par contre, au lieu de laisser le processeur empiler l'adresse de retour, cette dernière est définie statiquement à l'adresse *a0x`fffffe62`*. Après l'exécution de la routine à l'adresse `0xfffffd8d` l'exécution se poursuivra à l'adresse `*0xfffffe62 == 0xfffffea5`.

```
d.bios.rom:     format de fichier binary


Déassemblage de la section .data:

fffffea5 <.data+0x3ffea5>:
fffffea5:   0b c0                   or     %eax,%eax
fffffea7:   74 0c                   je     0xfffffeb5
fffffea9:   b9 79 00 00 00          mov    $0x79,%ecx
fffffeae:   33 d2                   xor    %edx,%edx
fffffeb0:   83 c0 30                add    $0x30,%eax
fffffeb3:   0f 30                   wrmsr
fffffeb5:   bf dc 80 0f f8          mov    $0xf80f80dc,%edi
fffffeba:   83 0f 08                orl    $0x8,(%edi)
fffffebd:   b9 a0 01 00 00          mov    $0x1a0,%ecx
fffffec2:   0f 32                   rdmsr
fffffec4:   0f ba f0 16             btr    $0x16,%eax
fffffec8:   73 02                   jae    0xfffffecc
fffffeca:   0f 30                   wrmsr
fffffecc:   b9 1b 00 00 00          mov    $0x1b,%ecx
fffffed1:   0f 32                   rdmsr
fffffed3:   83 e2 f0                and    $0xfffffff0,%edx
fffffed6:   25 ff 0f 00 00          and    $0xfff,%eax
fffffedb:   0d 00 00 e0 fe          or     $0xfee00000,%eax
```

```
fffffee0:   0f 30                   wrmsr
fffffee2:   0f 20 e0                mov    %cr4,%eax
fffffee5:   0d 00 06 00 00          or     $0x600,%eax
fffffeea:   0f 22 e0                mov    %eax,%cr4
fffffeed:   b0 03                   mov    $0x3,%al
fffffeef:   e6 80                   out    %al,$0x80
fffffef1:   ba 52 65 72 50          mov    $0x50726552,%edx
fffffef6:   b0 73                   mov    $0x73,%al
fffffef8:   e6 b2                   out    %al,$0xb2
fffffefa:   e6 84                   out    %al,$0x84
fffffefc:   e6 84                   out    %al,$0x84
fffffefe:   0a c0                   or     %al,%al
ffffff00:   75 0c                   jne    0xffffff0e
ffffff02:   66 ba f9 0c             mov    $0xcf9,%dx
ffffff06:   b0 02                   mov    $0x2,%al
ffffff08:   ee                      out    %al,(%dx)
ffffff09:   b0 06                   mov    $0x6,%al
ffffff0b:   ee                      out    %al,(%dx)
ffffff0c:   eb fe                   jmp    0xffffff0c
ffffff0e:   e9 dd fb ff ff          jmp    0xfffffaf0
```

TODO

Fonction `0xfffffd8d`

```
d.bios.rom:     format de fichier binary


Déassemblage de la section .data:

fffffd8d <.data+0x3ffd8d>:
fffffd8d:   b0 02                   mov    $0x2,%al
fffffd8f:   e6 80                   out    %al,$0x80
fffffd91:   bb 00 00 c2 ff          mov    $0xffc20000,%ebx
fffffd96:   8b d3                   mov    %ebx,%edx
fffffd98:   8b 43 30                mov    0x30(%ebx),%eax
fffffd9b:   25 ff ff 00 00          and    $0xffff,%eax
fffffda0:   03 d8                   add    %eax,%ebx
fffffda2:   03 52 20                add    0x20(%edx),%edx
fffffda5:   83 3b ff                cmpl   $0xffffffff,(%ebx)
fffffda8:   74 24                   je     0xfffffdce
fffffdaa:   b9 04 00 00 00          mov    $0x4,%ecx
fffffdaf:   8b f3                   mov    %ebx,%esi
fffffdb1:   bf 65 fd ff ff          mov    $0xfffffd65,%edi
fffffdb6:   f3 a7                   repz cmpsl %es:(%edi),%ds:(%esi)
fffffdb8:   74 1d                   je     0xfffffdd7
fffffdba:   8b 43 14                mov    0x14(%ebx),%eax
fffffdbd:   25 ff ff ff 00          and    $0xffffff,%eax
fffffdc2:   03 d8                   add    %eax,%ebx
fffffdc4:   83 c3 07                add    $0x7,%ebx
fffffdc7:   83 e3 f8                and    $0xfffffff8,%ebx
fffffdca:   3b da                   cmp    %edx,%ebx
fffffdcc:   72 d7                   jb     0xfffffda5
fffffdce:   b0 0e                   mov    $0xe,%al
fffffdd0:   e6 80                   out    %al,$0x80
fffffdd2:   e9 88 00 00 00          jmp    0xfffffe5f
fffffdd7:   8b fb                   mov    %ebx,%edi
fffffdd9:   8b 43 14                mov    0x14(%ebx),%eax
fffffddc:   25 ff ff ff 00          and    $0xffffff,%eax
fffffde1:   03 f8                   add    %eax,%edi
fffffde3:   83 c3 18                add    $0x18,%ebx
fffffde6:   8b f3                   mov    %ebx,%esi
fffffde8:   b8 01 00 00 00          mov    $0x1,%eax
fffffded:   0f a2                   cpuid
fffffdef:   8b d8                   mov    %eax,%ebx
fffffdf1:   b9 17 00 00 00          mov    $0x17,%ecx
fffffdf6:   0f 32                   rdmsr
fffffdf8:   c1 ea 12                shr    $0x12,%edx
fffffdfb:   80 e2 07                and    $0x7,%dl
fffffdfe:   8a ca                   mov    %dl,%cl
fffffe00:   b2 01                   mov    $0x1,%dl
fffffe02:   d2 e2                   shl    %cl,%dl
fffffe04:   3b f7                   cmp    %edi,%esi
fffffe06:   73 57                   jae    0xfffffe5f
fffffe08:   83 3e ff                cmpl   $0xffffffff,(%esi)
fffffe0b:   74 52                   je     0xfffffe5f
fffffe0d:   b9 00 08 00 00          mov    $0x800,%ecx
fffffe12:   83 7e 1c 00             cmpl   $0x0,0x1c(%esi)
```

```
fffffe16:   74 03                   je     0xfffffe1b
fffffe18:   8b 4e 20                mov    0x20(%esi),%ecx
fffffe1b:   3b 5e 0c                cmp    0xc(%esi),%ebx
fffffe1e:   75 07                   jne    0xfffffe27
fffffe20:   8b c6                   mov    %esi,%eax
fffffe22:   84 56 18                test   %dl,0x18(%esi)
fffffe25:   75 3a                   jne    0xfffffe61
fffffe27:   8b 6e 20                mov    0x20(%esi),%ebp
fffffe2a:   8b 46 1c                mov    0x1c(%esi),%eax
fffffe2d:   83 c0 30                add    $0x30,%eax
fffffe30:   3b e8                   cmp    %eax,%ebp
fffffe32:   76 1b                   jbe    0xfffffe4f
fffffe34:   8b 0c 30                mov    (%eax,%esi,1),%ecx
fffffe37:   8d 6c 30 14             lea    0x14(%eax,%esi,1),%ebp
fffffe3b:   39 5d 00                cmp    %ebx,0x0(%ebp)
fffffe3e:   75 07                   jne    0xfffffe47
fffffe40:   8b c6                   mov    %esi,%eax
fffffe42:   84 55 04                test   %dl,0x4(%ebp)
fffffe45:   75 1a                   jne    0xfffffe61
fffffe47:   83 c5 0c                add    $0xc,%ebp
fffffe4a:   e2 ef                   loop   0xfffffe3b
fffffe4c:   8b 4e 20                mov    0x20(%esi),%ecx
fffffe4f:   81 c1 ff 01 00 00       add    $0x1ff,%ecx
fffffe55:   81 e1 00 fe ff ff       and    $0xfffffe00,%ecx
fffffe5b:   03 f1                   add    %ecx,%esi
fffffe5d:   eb a5                   jmp    0xfffffe04
fffffe5f:   33 c0                   xor    %eax,%eax
```

['python3.2 ./disasm-to-graph.py -o 0xffc00000 -s 0xfffffd8d -e fffffe61 -b 32 -f d.bios.rom', 'dot -Tpng -o 0xfffffd8d.png'] .. image:: 0xfffffd8d.png

TODO