Phoenix®

Embedded Controller in an SBS enabled system. APM & ACPI

By Aleksandr Frid, Phoenix Technologies

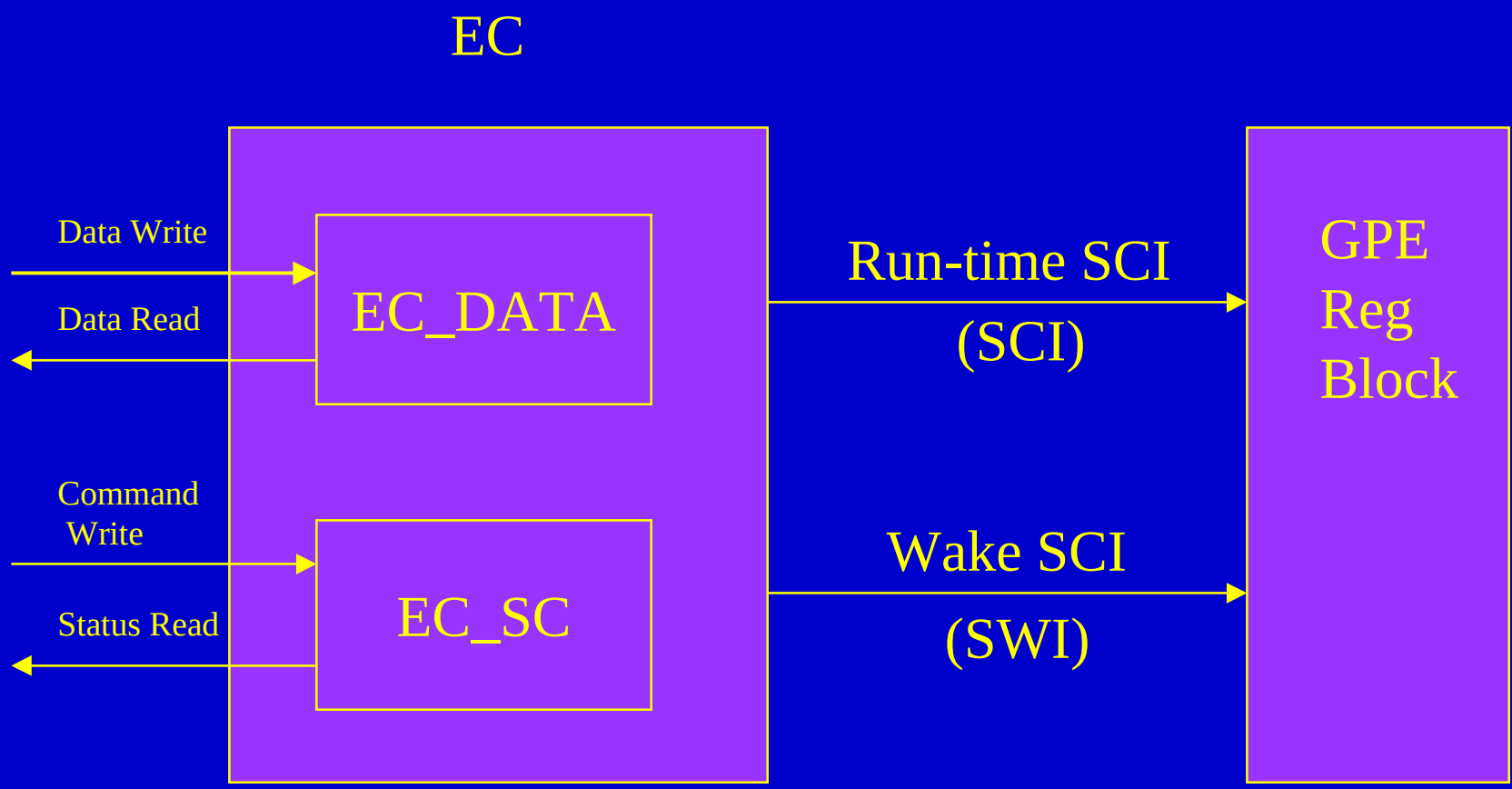
Topics

- ACPI EC interface overview
- SBS access via EC space
- SBS under APM and ACPI OS

ACPI EC Interface Overview

- EC Generic Model
- EC Host Interface
- EC Interrupts
- Basic EC Commands
- Event Management and Query Command

EC Generic Model



EC Host Interface

- The Data Port (e.g. 62h)
- The Command/Status Port (e.g. 66h)
  - B7 - Reserved
  - B6 - SMIEVT - 1= SMI Event Pending
  - B5 - SCIEVT - 1= SCI Event Pending
  - B4 - BURST - EC Burst Mode Enabled
  - B3 - C_D2 - Command / Data Flag
  - B2 - Reserved
  - B1 - IBF
  - B0 - OBF

EC Interrupts

- Run time SCI (SCI) is used
  - For Host Interface communications
  - To report Run-time Events
  - SCI is treated as an edge event, connected directly to GPE input with no debouncing.
- Wake SCI (SWI) is used
  - To report Wake-up events only
  - SWI can be shared, and can be treated as a level or edge event

Basic EC Commands

- Owned by EC driver
  - No direct port access from BIOS ASL code
  - Operation Regions in EC space
- 80h - Read Embedded Controller
- 81h - Write Embedded Controller
- 82h - Burst Enable Embedded Controller
- 83h - Burst Disable Embedded Controller
- IBF, OBF - Command Specific SCI Model

Event Management / Query Command

- Run-time events are reported via SCI
  - Each event has unique notification byte (ID)
  - Event Notification Queue in EC
  - SCIEVT - SCI Event pending - flag in status
- Handled by OS (EC driver)
  - Query EC command (84h) to get event ID XX
  - Invoke Control Method _QXX (BIOS)
- Wake-Up events are reported via SWI, they are handled by generic ASL Control Methods (BIOS)

SBS access via EC space

- EC SMBus registers set
- SMBus Events
- Direct OS access to SBS
- ACPI SBS description
- Non SBS Devices description
- Control Methods SBS access

EC SMBus Register Set

- 40 registers in EC space @ offset EC_SMB_Base (e.g. 18h)
  - EC_SMB_Base+00 = SMB_PRTCL
  - EC_SMB_Base+01 = SMB_STS
  - EC_SMB_Base+02 = SMB_ADDR
  - EC_SMB_Base+03 = SMB_CMD
  - EC_SMB_Base+04…35 = SMB_DATA[0…31]
  - EC_SMB_Base+36 = SMB_BCNT
  - EC_SMB_Base+37 = SMB_ALRM_ADDR
  - EC_SMB_Base+38,39 = SMB_ALRM_DATA[0,1]
- New commands is initiated when SMB_PRTC ! = 0 (only standard SMBus protocols are supported)

SMBus Events

- Command completion (successful or not)
- SMBus Messages
  - Smart Battery Alarm Message
  - SMBus Alert Signal from Smart Selector / Smart Charger
    - AC presence
    - Battery insertion and removal
    - Charging status changes (optional)
  - EC translates Alert Signal to SMBus Message and generates SCI

SMBus Events

- SMBus Events have common query value (e.g. 20h)
- SMB_STS
  - B7 - Command O.K.
  - B6 - Alarm
  - B5 - Reserved
  - B4-0 - Error code
- Only 1-Alarm buffer is available: no other message is accepted by EC until the previous one is serviced by the System (SMBus device must repeat discarded message)

Direct OS access to SBS

- SBS: batteries, selector and charger (may be no selector if only 1 battery)
- All devices supports ALL commands found in SBS specifications 1.0 (exception SB 0x00)
- SBS is behind ACPI compliant EC
- ACPI BIOS describes SBS object in ACPI name space
- No need for control methods to access SBS data and process SBS events

ACPI SBS Description

```
Device(EC0)
{       Name(_HID, EISAID("PNP0C09"))                   // EC PnP ID
        Name(_CRS, Buffer() {// EC I/O ports 62h,66h })
        Name(_GPE, 0)    // EC SCI is connected to GPE.0

        Device(SMB0)
        {       Name(_HID,"ACPI0001")           // SMBus Host ID
                Name(_EC, 0x1820)               // Offset and Query
                Device(SBS0)
                {       Name(_HID,"ACPI0002")// SBS ID
                        Name(_SBS, 0x02)        // 2 batteries+selector
                }
        }
}
```

Non SBS Devices Description

```
Device(EC0)
{       Name(_HID, EISAID("PNP0C09"))                   // EC PnP ID
        Name(_CRS, Buffer() {// EC I/O ports 62h,66h })
        Name(_GPE, 0)    // EC SCI is connected to GPE.0
        Device(SMB0)
        {       Name(_HID,"ACPI0001")           // SMBus Host ID
                Name(_EC, 0x1820)               // Offset and Query
                OperationRegion(SMR0, SMBus, 0x4F, 0x1)
                Device(TS0)
                {       Name(_ADR, 0x4F)        // Actual slave address
                        Field(SMR0, ByteAcc, NoLock, Preserve)
                        {       AccessAs(ByteAcc, 0),     //Prtcl +Cmnd
                                TEMP, 8         // Temperature register
                        }
                }
        }
}
```

Control Methods SBS access (no SMB OS drivers)

```
Device(EC0)
{       Name(_HID, EISAID("PNP0C09"))                   // EC PnP ID
        Name(_CRS, Buffer() {// EC I/O ports 62h,66h })
        Name(_GPE, 0)    // EC SCI is connected to GPE.0

        OperationRegion(ERAM, EmbeddedControl, 0, 0xff)
        Field(ERAM, ByteAcc, NoLock, Preserve)
        {
                Offset(0x18),
                SMPR,  8,               // SMB protocol register
                SMST,  8,               // SMB status register
                SMAD,  8,               // SMB address register
                SMCM, 8,                // SMB command register
                SMW0, 16,               // SMB data 0 and 1
        }
```

Control Methods SBS access (no SMB OS drivers)

```
Event(SMBE)                     // SMBus sync object

Method(RDCP)                    // Read Battery Capacity
{
        Store(0x0F, SMCM)   // 0Fh = read capacity
        Store(0x16, SMAD)   // Battery address
        Store(0x09, SMPR)   // Read word
        Reset(SMBE)
        Wait(SMBE, 0xFFFF)  //Wait cmd completion
        Store(SMW0, Local0)  // Read battery response
        Return(Local0)
}
```

Control Methods SBS access (no SMB OS drivers)

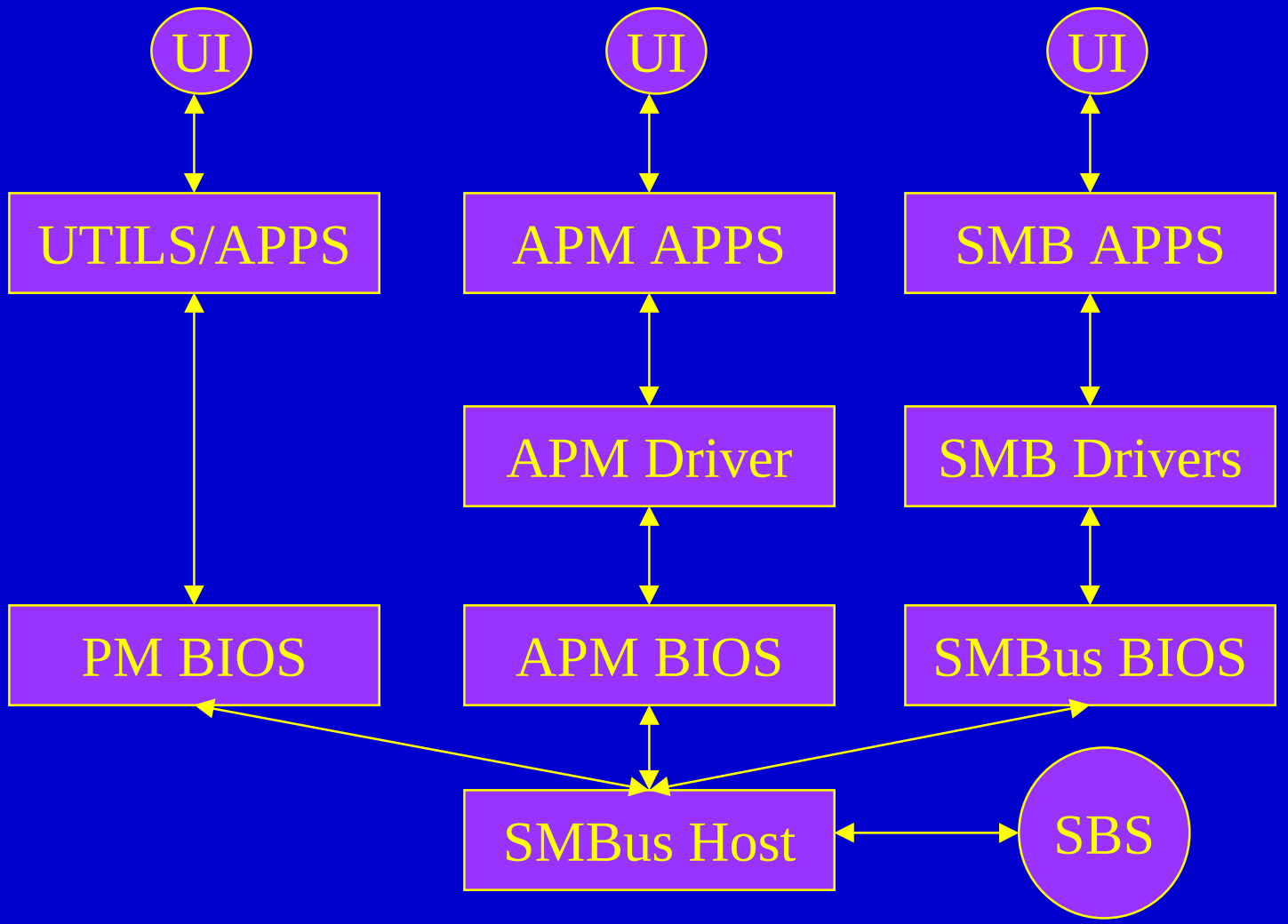
```
Method(_Q20)
{
        Signal(SMBE)        // signal cmd completion
}

} // end device EC0
```

Note. This code is just an illustration (it does not have error checking, retries, alarm verification ...)

SBS under APM and ACPI OS

- SBS support in APM environment
- SBS support in ACPI Smart Battery environment
- SBS support in ACPI Control Battery environment

SBS in APM environment

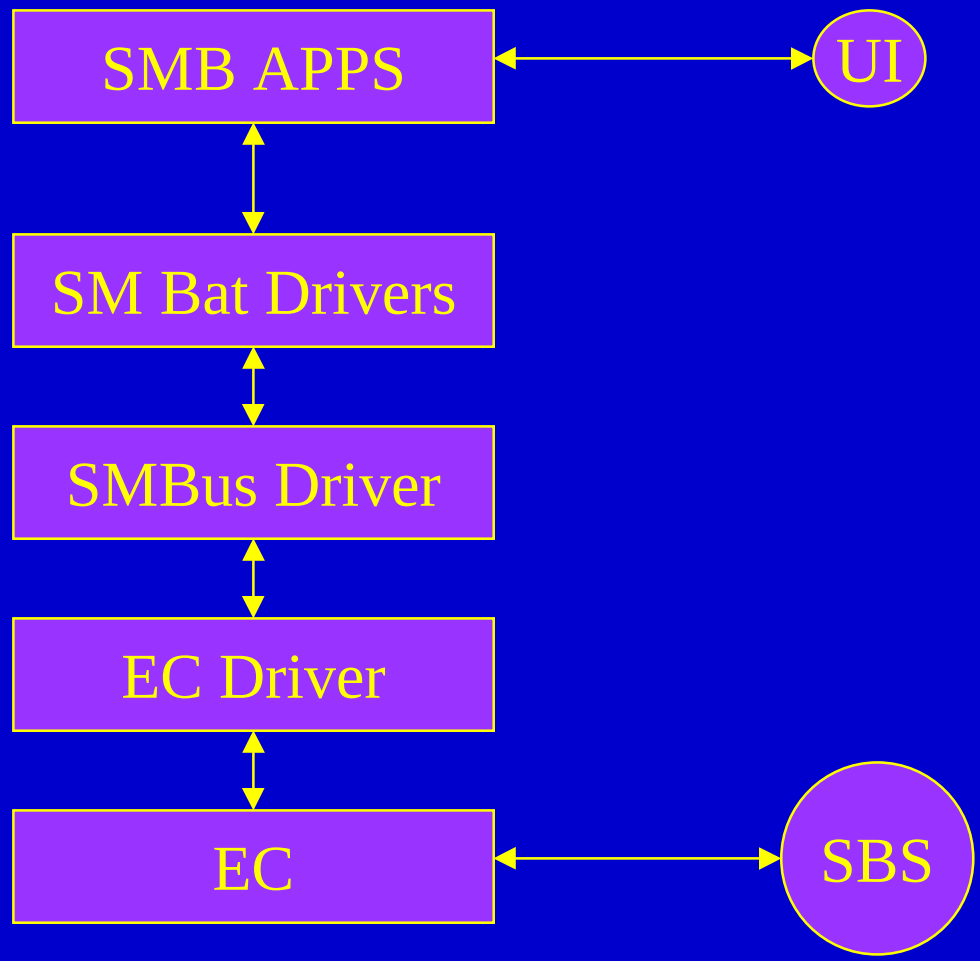


Embedded Controller in an SBS enabled system. APM & ACPI

SBS in APM environment

- SMBus Host - chipset or combined with KBC
- SMBus access synchronization in BIOS
- Drivers: APM (OS native), SMB
- Applications
  - SMB, e.g. Phoenix BatteryScope
  - APM Control Panel (OS native) or APM OEM
  - PM, e.g. PM Setup or Phoenix PowerPanel
- Multiple policies

SBS in ACPI SM Battery environment

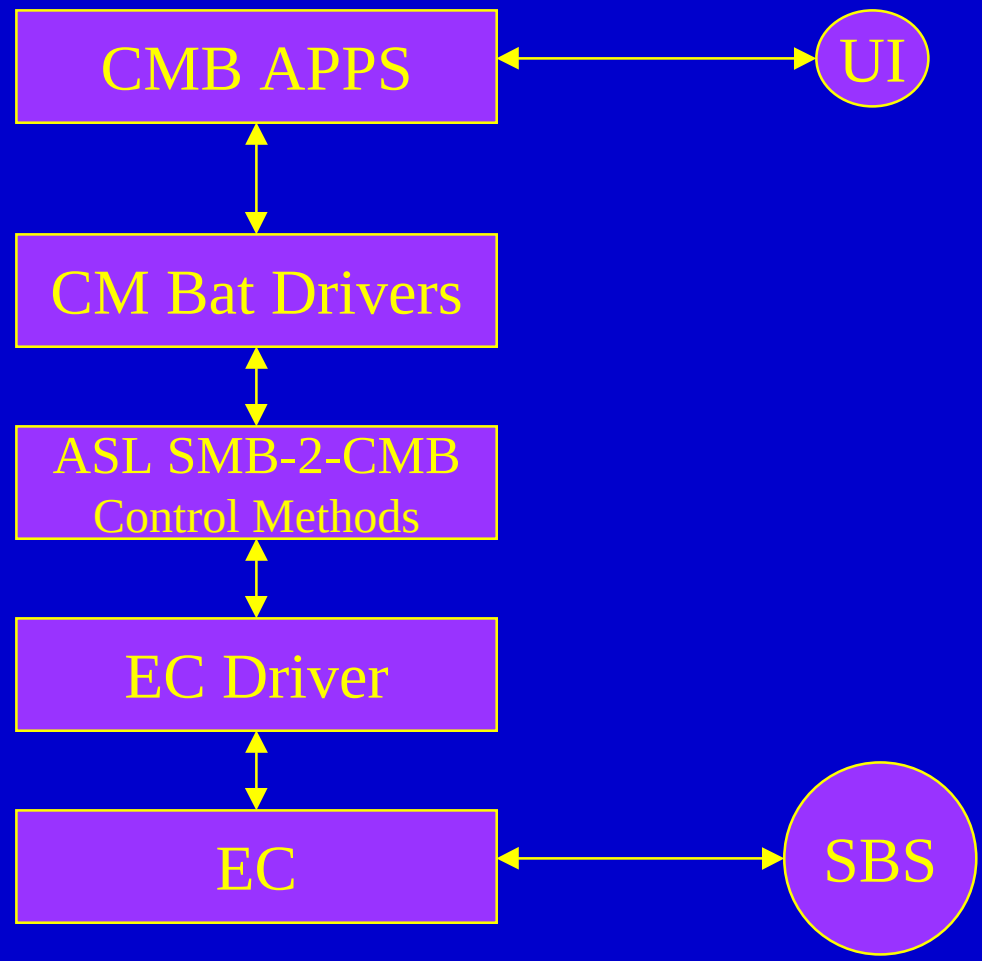


Embedded Controller in an SBS enabled system. APM & ACPI

SBS in ACPI SM Battery environment

- SMBus Host - EC (combined with KBC)
- SMBus access synchronization in OS (SMBus driver)
- All drivers are native OS drivers
- Applications
  - OS native, e.g. Power Meter
  - OEM
- One policy carried out by OS

SBS in ACPI CM Battery environment



SBS in ACPI CM Battery environment

- Main differences from SM Battery
  - SMBus access synchronization in ASL
  - Data limited to CM battery specification (for example no At Rate information is available)
- Looks the same for the end-user