DV 16 - Serial GPi (General-Purpose Input)

Devices

Written by: Craig Prouse February 1991

This Technical Note discusses the latest supported methods for reading, validating, and configuring the GPi serial input across all members of the Macintosh family.

GPi is a software-configurable serial input present on some machines. It is located at pin 7 on the DIN-8 serial connectors, and connects to the DCD input of the Z8530 Serial Communications Controller (SCC). Because DCD is monopolized by the mouse on the Macintosh Plus, GPi is not implemented on that machine. Other machines which do not support GPi include the Macintosh Classic and Macintosh LC. On these machines, pins 7 of the DIN-8 serial connectors are not connected.

Reading GPi (The Easy Part)

A number of developers currently make use of the GPi input on the serial ports of the Macintosh SE, Macintosh II, and Portable families. It's a handy feature and DTS regularly receives the question of how to read this input. The code required is actually quite simple, assuming all the proper hardware support is in place. As stated previously, some Macintosh models do not support GPi. For those machines which do support GPi and for which the SCC chip is directly accessible, the following code reads the state of GPi.

```
        movea   (SCCRd).w,a0    ; best place to get address of SCC RR0
        move.b  aCtl(a0),d0     ; modem port--use bCtl for printer port
        btst    #3,d0           ; GPi comes in DCD input--bit 3 of SCC RR0
        beq     @GPi0
GPi1    ...
        ...
GPi0    ...
```

This is currently the only way to determine the state of the GPi serial input. There is no support for this signal in the Serial Driver. If the SCC is not directly accessible, then neither is GPi. To determine if the SCC is accessible, check with `_Gestalt`. If an SCC exists but is not accessible, `_Gestalt` claims that there is no SCC.

Validating and Configuring GPi (A Little Bit Harder)

To aid application developers in determining whether a machine supports GPi, a `_Gestalt` selector is available in System 6.0.7 and later. This selector is fully documented in *Inside Macintosh*, Volume VI, and specifies (a) whether GPi is supported on port A, (b) whether GPi

is supported on port B, and (c) whether GPi may be used as a clock input for synchronous modems on port A.

There is another new call which developers can use to configure GPiA as an external clock. Previously, developers had to manipulate a bit in VIA1 to enable or disable external clocking on this pin. Unfortunately, there has always been some ambiguity about the sense of this bit (the SE uses the opposite sense of the Macintosh II) and the VIA bit is not present at all on the Macintosh IIfx—see Technical Note HW 9 - Macintosh IIfx: The Inside Story. The friendly way to configure GPiA uses `_HwPriv` selector 7, as documented in that Technical Note.

MPW has never defined a high-level calling interface to this particular trap macro, and no glue has ever been available for Pascal and C programmers. Until this is remedied, the following inline glue fills in quite nicely:

```
FUNCTION SwapSerialClock (clock, portID: Integer) : Integer;
  INLINE $205F, $7007, $A198, $6B02, $3008, $3E80;

pascal short SwapSerialClock (short clock, short portID) =
{
  0x205F, 0x7007, 0xA198, 0x6B02, 0x3008, 0x3E80
}
```

For the normal 3.672 MHz internal serial clock, pass `$0000` in the `clock` parameter. For external clocking provided at the GPiA pin, pass `$0001` in the `clock` parameter. Other clock sources are theoretically possible, so use only one of these two values.

Only one value is currently supported for the `portID` parameter, and that is the Serial Driver enumerated constant `sPortA`. If necessary, this constant must be casted to type `short` or coerced to type `Integer`, according to the terminology of your development language.

If an error results, `SwapSerialClock` returns a negative number, otherwise it returns the previous GPiA configuration which is a non-negative number. This makes it convenient to save and restore the original state.

`SwapSerialClock` works with system software back to 6.0.5, although it does not achieve the desired results on the Macintosh IIfx. In fact, it may crash. This is a problem which is addressed in System Software 7.0. All the features described in this Note are technically new features for System 7.0, but Apple encourages developers to employ them if necessary (and available) in 6.0.x-compatible applications and suggest to their customers to use the latest available system software to obtain maximum benefit from these types of applications.

The following code fragment shows how to use these new features without explicitly depending upon specific system software versions. It assumes only that the `_Gestalt` trap is implemented or emulated by MPW glue (which is already available). It is not necessarily possible to trap the error of calling `SwapSerialClock` on a Macintosh IIfx with pre-7.0 software. It is best to avoid executing this code at all on such a configuration or else risk a system crash.

```
PROGRAM SerialClock;

  USES Types,GestaltEqu,Serial;
```

```
CONST
  internalClock = 0;     { convenient constants for SwapSerialClock }
  externalClock = 1;

VAR
  gestErr    : OSErr;
  hasGPiAClk : Boolean;
  oldClockMode: Integer;
  result     : LongInt;

FUNCTION SwapSerialClock(clock,portID: Integer): Integer;
  INLINE $205F,$7007,$A198,$6B02,$3008,$3E80;
{ this could be supported in a future version of MPW }

BEGIN
  gestErr := Gestalt(gestaltSerialAttr,result);
  IF gestErr = noErr THEN BEGIN
    hasGPiAClk := (band(result,bsl(1,gestaltHasGPIaToDCDa)) <> 0);
    IF hasGPiAClk THEN BEGIN
      { SwapSerialClock is supported if gestaltHasGPIaToDCDa is supported }
      { it may experience difficulties with Mac IIfx and pre-7.0 systems… }
      oldClockMode := SwapSerialClock(internalClock,Integer(sPortA));
      IF oldClockMode < 0 THEN BEGIN
        { handle case of error setting the clock mode }
      END;
    END
    ELSE BEGIN
      { handle case where there is no GPiA clock support }
    END;
  END
  ELSE BEGIN
    { handle case where Gestalt doesn't know about serial attributes }
    { this usually means assume no support, or ask for later system… }
  END;
END.
```

Further Reference:

- *Inside Macintosh*, Volume III, The Macintosh Hardware
- *Inside Macintosh*, Volume VI, Compatibility Guidelines
- *Guide to the Macintosh Family Hardware*, Serial I/O Ports
- Technical Note OV 16 - Gestalt and SysEnvirons : a Never Ending Story
- Technical Note HW 9 - Macintosh IIfx: The Inside Story
- Technical Manual: *Z8530 SCC Serial Communications Controller* (contact Zilog or AMD)