# New Technical Notes

## Macintosh

®

Developer Support

## High-Level Control and Status Calls: When a Good Call Goes Bad
**Devices**

Revised by: Gordon Sheridan                                          January 1992
Written by: Tim Enwall                                               February 1990

This Technical Note discusses situations under which high-level `Status` calls do not work correctly and `PBStatus` calls should be made instead.
**Changes since February 1990:** Information has been added describing similar problems with high-level `Control` calls.

---

When Apple designed the `_Control` and `_Status` traps, it was assumed that `_Control` would be used to send the driver a command or information and `_Status` would be used to request information from the driver. The `csParam` parameter was meant to pass information in only one direction for each call; **to** a driver on control calls and **from** a driver on status calls. Most drivers follow this convention, and so there is no problem using the high-level `Control` and `Status` calls with them. However, some drivers depend on the bidirectional transfer of information on `_Control` and `_Status` calls through the `csParam` variable. For drivers of this type it behooves you to always use the `PBControl` and `PBStatus` calls. It's not hard. Honest!

Glue code is used to build the parameter block that gets passed to the `_Control` and `_Status` traps. The glue uses the `refNum` and `csCode` parameters to fill out the corresponding fields in the parameter block.

On control calls it copies the data pointed to by `csParamPtr` into the `csParam` field of the parameter block before it calls `_Control`, but it does **not** copy the information back **after** the call.

On status calls it only copies the `csParam` field to the location pointed to by `csParamPtr` after the `_Status` call. It does **not** copy the data pointed to by `csParamPtr` into the parameter block **before** calling `_Status`.

The low-level `PBControl` and `PBStatus` calls have no such problem because you are working directly with the parameter block and have direct access to the `csParam` field. The high-level `Control` and `Status` calls in some cases either work incorrectly, or worse,

cause problems for the device driver.

An example of _Control calls that return data in csParam field are disk drivers that return a pointer to their icon on _Control calls with a csCode of 21. Here is an example of how you can use PBControl to get the icon of a drive and add it to the current resource file.

```
#define             kGimmeIconDataPtr         21
#define             kSizeOfIconAndMask  256
#define             kCustomAliasIconID  128

AddVolumeIconRes (short  vRefNum)
{
      HVolumeParam  vInfoPB;
      CntrlParam    cntlPB;
      Handle              serverIcon;
      OSErr               err;

      vInfoPB.ioNamePtr       = nil;
      vInfoPB.ioVRefNum       = vRefNum;                  // can be working
                                      // directory
      vInfoPB.ioVolIndex  = 0;                            // use vRefNum -
                                      // don't index

      err = PBHGetVInfoSync ((HParamBlockRec *) &vInfoPB);

      if (err == noErr)
      {
            cntlPB.ioVRefNum    = vInfoPB.ioVDrvInfo;     // logical drive
                                      // number
            cntlPB.ioCRefNum    = vInfoPB.ioVDRefNum;     // disk driver
                                      // reference number
            cntlPB.csCode       = kGimmeIconDataPtr;

            err = PBControl ((ParamBlockRec *) &cntlPB, false); // false =
                                            // synchronous

            if (err == noErr)                                               //
copy ICN# and add to resource fork
            {
                  serverIcon = NewHandle (kSizeOfIconAndMask);
                  if (serverIcon != nil)
                  {
                        BlockMove (* (Ptr *) (cntlPB.csParam), *serverIcon,
                              kSizeOfIconAndMask);
                        AddResource (serverIcon, 'ICN#', kCustomAliasIconID,
                  nil);
                        if (ResError () != noErr)
                              DisposHandle (serverIcon);
                  }
            }
      }
}
```

The most obvious example of a device driver that expects csParam as input on a _Status call is the video device driver(s) for Macintosh II Video Cards. Almost all of the documented status calls require csParam to point to some kind of table. In this case, most of the device driver's status routines do not function properly **if** using the high-level Status call.

Therefore, if you are interfacing to a device driver that you either know or suspect requires csParam for its status calls, use the low-level PBStatus call instead of the high-level

`Status` call. Likewise, if the driver returns information via the csParam field on control calls, you will need to use `PBControl` rather than the high-level `Control` call.

If you are writing a device driver, alert the users of your driver to these limitations. Alternatively, you could design your driver so that control calls only **receive** data and status calls only **return** data in the `csParam` field.

**Further Reference:**
- • *Inside Macintosh*, Volume II, The Device Manager
- • *Inside Macintosh*, Volume IV, The Disk Driver
- • *Inside Macintosh*, Volume V, The Disk Driver