# Four Can Play–Supporting Parallel Port Joysticks

## by Chris Ludwig

One of the many reasons people buy computer-based products is to use them as an ‘‘entertainment box’’. Amiga users are no exception in this regard. The Amiga has all the ingredients needed to be a great game machine; fast high-quality graphics, high-fidelity stereo sound, fast processor, and built-in joystick ports.

With this level of hardware support, it’s no wonder that some of the industry’s best games continue to be written for the Amiga first.

The only kind of games that don’t currently lend themselves to the Amiga’s built-in hardware are those that are meant to be played by more than two players simultaneously. Several types of games fall into this category, the most notable being the scrolling cooperative. This type of game, where up to four players control on-screen versions of themselves, is quite popular in video arcades. Another format that lends itself to having more than two joysticks is the board game simulation. It’s simply easier to play a four player board game if each player has his or her own control.

Because the Amiga has only two joystick ports, there are currently only a small number of ‘‘greater than two’’ player games available. So how do these games get around the two port limitation? The developers of these games have worked up a little bit of external hardware that remaps some parallel port input lines as two more joystick ports for single button joysticks.

Fortunately, these developers are all using the same wiring in their interfaces. Unfortunately, neither the interface nor its specifications have been widely available.

This article explains the construction of these parallel port joystick adapters (this article refers to them ‘‘+2’’ adapters), and demonstrates a coding strategy which will allow your software to access these two new ports. Though the instructions are fairly simple, if not done properly, it is possible to damage your computer, so Commodore does not recommend that you build these yourself. This information is provided for your convenience and is expressly subject to the disclaimer and warnings found at the beginning of this publication.

| Parallel function | Parallel port (DB-25 male) | JOY3 (DB-9) | JOY4 (DB-9) | Joystick function |
|---|---|---|---|---|
| Data bit 0 | 2 | 1 | | JOY3 UP |
| Data bit 1 | 3 | 2 | | JOY3 DOWN |
| Data bit 2 | 4 | 3 | | JOY3 LEFT |
| Data bit 3 | 5 | 4 | | JOY3 RIGHT |
| Data bit 4 | 6 | | 1 | JOY4 UP |
| Data bit 5 | 7 | | 2 | JOY4 DOWN |
| Data bit 6 | 8 | | 3 | JOY4 LEFT |
| Data bit 7 | 9 | | 4 | JOY4 RIGHT |
| Busy | 11 | | 6 | JOY4 FIRE |
| Select (Online) | 13 | 6 | | JOY3 FIRE |
| ground | 18 | | 8 | JOY4 GROUND |
| ground | 19 | 8 | | JOY3 GROUND |

**The ‘‘+2’’ Interface**

## Building the ‘‘+2’’ Interface

The interface that these two extra joystick ports require is quite straightforward in design and does not require any power. Essentially, two male 9-pin ‘‘d-sub’’ connectors (the type commonly used for joystick ports) are wired directly to a 25-pin ‘‘d-sub’’ connector.

The gender of the 25-pin connector depends on the Amiga model. Amiga 1000 computers have a non-standard, male 25-pin parallel port, so the Amiga 1000 adapters will need a female connector. All other Amigas have standard female 25-pin parallel ports, so adapters for A500, A600, A2000, A3000, A3000T, and CDTV units should have a male 25-pin connector.

The only difference between the requirements for the A1000 ‘‘+2’’ adapter and the ‘‘+2’’ adapter for the rest of the Amiga/CDTV product line is the gender. The ‘‘+2’’ adapter only uses parallel port pins that all Amigas/CDTVs have in common. These particular pins were chosen so that one ‘‘+2’’ adapter and a gender changer could connect two joysticks to any existing Amiga. Because some of the unused pins carry voltages, *only wire the pins specified in the chart! Doing otherwise can destroy components inside the Amiga!*

The ‘‘+2 Interface’’ chart above contains a list of the pins which need to be connected to create the ‘‘+2’’ adapter.

| Parallel function | Parallel port (DB-25 male) | JOY3 (DB-25 male) | JOY4 | Joystick function |
|---|---|---|---|---|
| Data bit 0 | 2 | 1 | | JOY3 UP |
| Data bit 1 | 3 | 2 | | JOY3 DOWN |
| Data bit 2 | 4 | 3 | | JOY3 LEFT |
| Data bit 3 | 5 | 4 | | JOY3 RIGHT |
| Data bit 4 | 6 | | 9 | JOY4 UP |
| Data bit 5 | 7 | | 10 | JOY4 DOWN |
| Data bit 6 | 8 | | 11 | JOY4 LEFT |
| Data bit 7 | 9 | | 12 | JOY4 RIGHT |
| Busy | 11 | | 22 | JOY4 FIRE |
| Select (Online) | 13 | 14 | | JOY3 FIRE |
| ground | 18 | | 24 | JOY4 GROUND |
| ground | 19 | 16 | | JOY3 GROUND |

**The Non-Production ‘‘+2’’ Interface**

## Housing the Interface

For development purposes, you can construct a ‘‘+2’’ adapter from two 25-pin male d-sub connectors. One connector attaches to the parallel port, while the other accepts the two 9-pin female joystick plugs. Simply build the ‘‘number 3’’ joystick connector into the leftmost 9 pins, and the ‘‘number 4’’ connector into the rightmost 9 pins. The pin-outs for this setup are in the ‘‘Non-Production +2 Interface’’ chart above.

This arrangement is attractive to developers primarily because it has few parts and it’s easy to construct. It is appropriate only for developer use and testing. Do not consider it for any sort of user distribution, because the connectors for certain brands of joysticks are too big to fit side by side into a 25-pin connector. Users will become quite irate upon finding that they can’t use their ‘‘wiz-stik 5000’’ with their 4 player game adapter.

With this in mind, adapters intended for use by end users should be built into some sort of casework. The box should include two suitably spaced joystick connectors, and a cable that allows the user to easily attach the adapter to the parallel port. Be sure that the cable is long enough to allow users to place the adapter where they can get to it easily.

Alternatively, the device could simply be a pair of long cables in a ‘‘Y’’ shape, with a parallel connector at the base of the ‘‘Y’’, and joystick connector on the other ends. Such an adapter would probably be the cheapest way to go.

Developers must caution end users to turn their computer's power off before plugging in a ''+2'' adapter. This will prevent users from accidentally destroying the 8520 chips which control the parallel port. To further prevent 8520 damage, developers should choose 25-pin connectors without a metal case or shield. While plugging the device in, it's rather easy to accidentally brush (and short) the metal shields of certain connectors against the pins of the parallel port.

No matter what the decision regarding implementation, remember that game players can be quite rough. All cables should be suitably long, and all connectors should utilize a proper strain relief system.

## Supporting the Interface in Software

Because this ''+2'' joystick adapter attaches to the parallel port, supporting it in software is simply a matter of reading the correct lines on the parallel port.

Many Amiga game programmers make the decision to throw away the OS and directly manipulate the hardware. These programmers will no doubt come up with their own personal scheme for reading the appropriate information from the port. Other, more forward thinking developers, may want to write their software in such a way as to be compatible with the OS.

The following pair of code samples work in tandem to demonstrate one method of reading the necessary information from the parallel port in an OS friendly manner. The C program *4play.c* is a simple example that demonstrates how to access the assembly functions. The assembly language program properly allocates the parallel port and any necessary signal lines, warning the OS not to let any other applications use them. Other assembly routines do the actual hardware level reading and place the joystick values into C variables. A cleanup routine releases the port and signal lines. The example does not mask out the directional or fire button bits, it only prints the raw joystick data. I'll leave it as an exercise for the reader to interpret the meaning of the raw data.