Using the Buddy System with any Microcomputer Family or Brand

If a microcomputer or microcontroller can toggle an LED it can probably talk to a Buddy System unit, given the right routines.

Most manufacturers will support their entire family of micros quite extensively, and one of the more popular means of support is through serial communications interface applications, or serial data routines.

Whether they call them RS-232 or not, asynchronous serial data transfer routines conforming to N-8-1 and having a start bit and a stop bit and eight bits of data in between, are probably RS-232 routines in disguise.

Especially revealing are discussions of baudrates, which usually note the standards like 2400, 4800, 9600, 19.2K, and above, and usually end up specifying baudrate as being 1/(bit period).

These and others like them are all indications that the serial interface method employed for one-pin asynchronous communications is the same as RS-232, or so similar to it as to be interchangeable.

The major difference is that microcomputers using 5 volt supply levels routinely do not generate the + and – 12 VDC line driving levels of a bona-fide RS-232 system. This weakens the error immunity of the link, but does not otherwise eliminate any usefulness. For short runs, especially on the same pcb, 5V serial is just fine.

The BASIC Stamp and the Buddy System both use Microchip Technologies PIC16C5X family of microcomputers, with custom firmware in each. Both these 5V devices use serial signaling that conforms only to the RS-232 standard for content and logic, but not for drive voltages.

Most other manufacturers also support the 5V method of serial transfer, and leave it to the user to add line drivers and line receivers, and to build additional power supplies and circuitry, if and when deemed necessary.

However, for the sake of simple data transfer, the protocol itself serves well, even without the more elaborate electrical requirements. As a result, many microcomputer manufacturers supply Application Notes for their various families to support asynchronous serial transfer.

On the website you will find links on the Applications page to other manufacturers who supply serial routines which allow RS-232 type data

transfers. By utilizing these routines in your particular model microcomputer, a generic capability to send and receive serial data in byte form on a single port-pin becomes available to you. Once you have this ability, it becomes trivial to use the Buddy System unit with your particular brand of microcomputer.

The commands are the same as those listed in The Buddy System Primer. All that would change is the words 'serin' and 'serout' no longer are used, nor is the argument 'baudrate' required, since the user would then control the baudrate with the resident application note previously discussed.

Finally, the user will already have set the port-pin to be used, within the confines of the application note routine itself. Once the direction and speed and pin are determined, the remaining command sequence to talk to the Buddy System consists merely of the string of bytes just as they are described in the Buddy System Primer, minus the 'BASIC Stamp specific' parts in the beginning of each command. It's merely a string of 'just-so-many-bytes-in-a-row,' which need to be sent out over the single pin.

Of course, it still remains for the user to organize their environment, such as including appropriate memory locations for data and string storage, and assigning variables for routine needs. As long as the user has a clear picture of where the data originates, and where incoming data is supposed to end up, everything should flow just as with the Stamp.

Higher-level programming languages, even Visual Basic, usually contain some serial I/O capability; how well implemented and useful it might be is another story. But within such an environment, once the baudrate, I/O pin, and direction are set, what remains is simply to pass the instruction sequence to the Buddy System as just so many bytes in a row…done!

Similarly, for incoming data, the user sets receive mode, a baudrate, and a pin-assignment, then expects to see just so many bytes in a row come marching in at the appointed time and baudrate…done!

Whether the user programs their microcomputer in Assembly Language, or using some higher-level environment, it is still possible to communicate to the Buddy System quite easily. See the links on the website for a listing of serial communications applications notes from various manufacturers, in particular, sending and receiving asynchronous serial data on one-wire.