

Parallel I/O Controller

MKB3881(P)-80/84

FEATURES

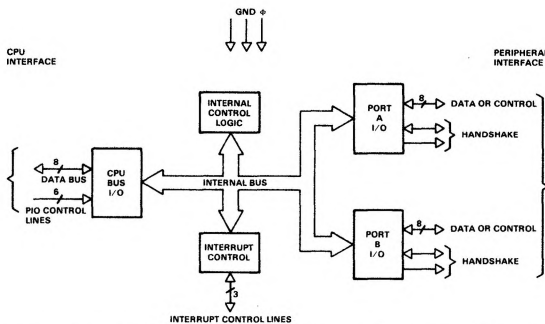
- Two independent 8 bit bidirectional peripheral interface ports with 'handshake' data transfer control
- Interrupt driven 'handshake' for fast response
- Any one of four distinct modes of operation may be selected for a port including:
 - Byte output
 - Byte input
 - Byte bidirectional bus (Available on Port A only)
 - Bit control mode
 All with interrupt controlled handshake
- Daisy chain priority interrupt logic included to provide for automatic interrupt vectoring without external logic
- Eight outputs are capable of driving Darlington transistors
- Typical ordering: MKB3881P-80 2.5 MHz Z80-PIO
MKB3881P-84 4.0 MHz Z80-PIO
- Single 5 volt supply and single phase clock required
- Fully processed to MIL-STD-883 Method 5004, Class B.
-55°C to 125°C temperature range
- MKI3881 available for Industrial/Hi-Rel users

DESCRIPTION

The Z80 Parallel I/O Circuit is a programmable, two port device which provides a TTL compatible interface between peripheral devices and the Z80-CPU. The CPU can configure the Z80-PIO to interface with a wide range of peripheral devices with no other external logic required. Typical peripheral devices that are fully compatible with the Z80-PIO include most keyboards, paper tape readers and punches, printers, PROM programmers, etc. The Z80-PIO utilizes N channel silicon gate depletion load technology and is packaged in a hermetic 40 pin DIP.

One of the unique features of the Z80-PIO that separates it from other interface controllers is that all data transfer between the peripheral device and the CPU is accomplished under total interrupt control. The interrupt logic of the PIO permits full usage of the efficient interrupt capabilities of the Z80-CPU during I/O transfers. All logic necessary to implement a fully nested interrupt structure is included in the PIO so that additional circuits are not required. Another unique feature of the PIO is that it can be programmed to interrupt the CPU on the occurrence of specified status conditions in the peripheral device. For example, the PIO can be programmed to interrupt if any specified peripheral alarm conditions should occur. This interrupt capability reduces the amount of time that the processor must spend in polling peripheral status.

PIO BLOCK DIAGRAM



PIO PIN CONFIGURATION

