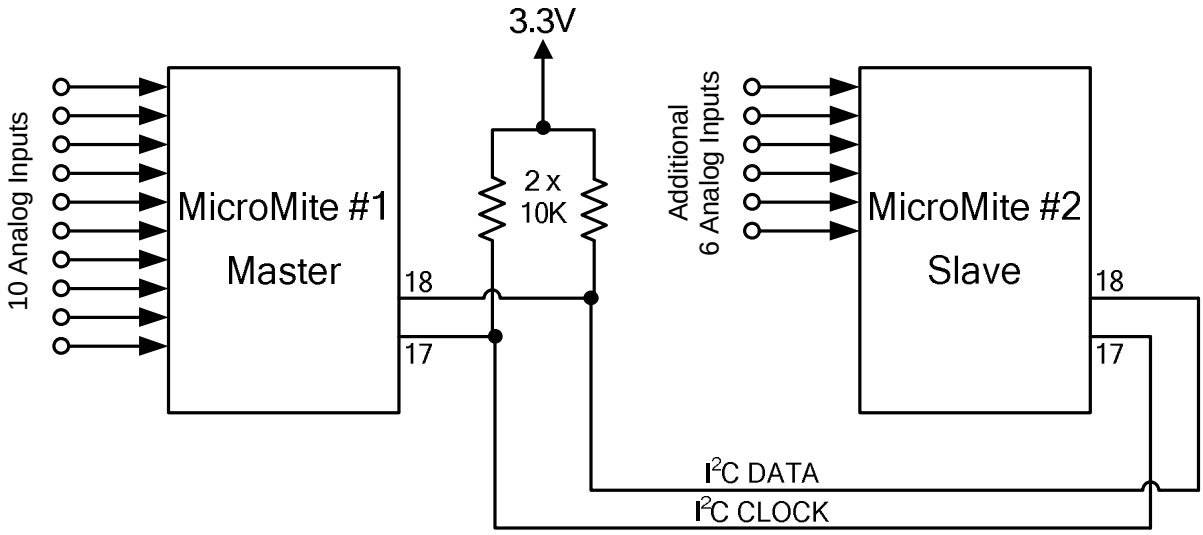
Example #2

There might be an occasion when a single MicroMite does not have enough serial ports, I/O pins, or whatever for a particular application. In that case a second MicroMite could be used as a slave to provide the extra facilities. Communications between the two is best implemented via I^2C.

For example, if sixteen analog inputs were required in a particular application a slave MicroMite could be programmed to read six of its analog inputs and transmit the data to the master via I^2C. The connections required are quite easy to implement and are illustrated below:



Program Running On the Slave:

The slave must first set up its I^2C interface to respond to requests from the master. With that done it can then drop into an infinite loop while the job of responding to the master is handled in the I^2C interrupts.

In the program below the slave expects to receive a one byte command from the master, which is the number of the input pin to be read. The slave then will wait for a request for data from the master. When that is received it will respond by measuring the voltage on the specified pin, converting that to a 12 character string and then sending the string off to the master.

This is the full program running on the slave:

```
For i = 2 To 7 : SetPin i, AIN : Next ' configure the inputs (pins 2 to 7)
I2C Slave Open 12, 0, 0, WriteD, ReadD' this slave's address is 12

Do:Loop                               ' do nothing - respond to interrupts

ReadD:                                ' received data from the master
  I2C Slave Read 1, pinnbr, t         ' get the pin number
  IReturn                             ' return from the interrupt

WriteD:                               ' received request from the master
  t$ = Str$(Pin(pinnbr))              ' measure the voltage on the pin
  t$ = t$ + Space$(12 - Len(t$))      ' pad it out to 12 characters
  I2C Slave Write 12, t$              ' send it to the master
  IReturn                             ' return from the interrupt
```

Program Running On the Master:

To get some data from the slave the master needs to send a one byte message to the slave (which is the pin number to measure) and then ask slave to send the data. The received data (which is a string of 12 characters) is saved to the array `data()`, so the master then needs to convert the individual characters to a string and then to a floating point number.

For convenience the routine has been encapsulated in a function. All that the program on the master need do is use this function (`GetSlaveInput(pinnbr)`) to read the voltage on the slave MicroMite's input.

This fragment demonstrates what is required:

```
Print "Input 2 is:" GetSlaveInput(2)  ' read pin 2 and display the result

Function GetSlaveInput(pinnbr)
  Local data(12), t$, i
  I2C Open 100, 1000                  ' speed is 100KHz, timeout is 1 sec
  I2C Write 12, 0, 1, pinnbr          ' send the pin number to address 12
  I2C Read 12, 0, 12, data(0)         ' and get the result back
  I2C Close
  For i = 0 To 11
    t$ = t$ + Chr$(data(i))           ' convert it to a string
  Next i
  GetSlaveInput = Val(t$)             ' and get the number
End Function
```

There are many features that could be added to these routines (eg, error checking, additional duties for the slave, accessing other inputs on the slave, etc) however this example provides a good start. If even more inputs were required then other MicroMite's could be connected to the same I2C bus (with different I^2C addresses) to provide almost unlimited expansion capability.