UNIT 11:
DATA TRANSFER
TRANSFER LOGIC

Modularization is necessary to keep complex designs at a manageable size, as well as to maintain efficiency in the debugging, testing and validation processes.

Synchronous design together with register/bus design is one way to accomplish modularization. Synchronous logic provided the isolation needed for independent design of individual modules. As along as the system clock runs sufficiently slowly, timing errors can be avoided.
TRI-STATE LOGIC

Digital circuits operate on values of logic one (high) and logic zero (low). However, current has to flow for the associated voltage levels to be realized. Hence, the need for "pull-up" or "pull-down" resistors.
TRI-STATE LOGIC

Digital circuits operate on values of logic one (high) and logic zero (low). However, current has to flow for the associated voltage levels to be realized. Hence, the need for "pull-up" or "pull-down" resistors.

1. $R_1$ determines how much current flows through the button when it is pressed closed. (Don't want too low, as power loss is $i^2R$; start at around 10 kΩ.)
2. $R_1$ determines, in conjunction with input impedance of op-amp, how much current flows into op-amp when circuit is open. (Don't want too high; normally choose order of magnitude less than input impedance of next stage.)
Two-state output

Open circuit allows next stage to sink a small amount of current (< 2 mA for TTL) to ground through resistor; results in low voltage (< 0.8 V) at output.

Tri-state output

In third state, no current flows through resistor; output neither high nor low. No influence on other circuitry.
TRI-STATE INVERTER

Logically
\[ A \quad \overline{A} \quad \text{EN} \]

Electrically
\[ V_{DD} \quad V_{SS} \quad \text{OUT} \]

Truth Table
<table>
<thead>
<tr>
<th>EN</th>
<th>A</th>
<th>OUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Z</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Z</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
REGISTERS

- Registers isolate sections (modules) of the system. They allow changes to be made at one section without affecting others.
- Used to store information temporarily, often as a step in transferring data to other modules.
- Usually referenced by number of bits that unit can hold; for example, an "8-bit register."
- Tri-state output commonly implemented.
REGISTERS

A register can be thought of as a set of D flip-flops:

- Each D flip-flop represents one bit
- N-bit register has N data signals and 2 or 3 control signals.
  
  - First control signal moves data into register (often denoted as L for latch or load)
    - Can be ANDed with the master clock to control data input.
  
  - Second signal is output-enable (OE) for the tri-state output.
    - Level triggered response. Output is connected when OE is TRUE.

  - Additional controls may be available to clear (reset) the register.
DATA REGISTER

- Choice of Eight Latches or Eight D-Type Flip-Flops in a Single Package
- 3-State Bus-Driving Outputs
- Full Parallel Access for Loading
- Buffered Control Inputs
- Clock-Enable Input Has Hysteresis to Improve Noise Rejection (‘S373 and ’S374)
- P-N-PInputs Reduce DC Loading on Data Lines (‘S373 and ’S374)

**description**

These 8-bit registers feature 3-state outputs designed specifically for driving highly capacitive or relatively low-impedance loads. The high-impedance 3-state and increased high-logic-level drive provide these registers with the capability of being connected directly to and driving the bus lines in a bus-organized system without need for interface or pullup components. These devices are particularly attractive for implementing buffer registers, I/O ports, bidirectional bus drivers, and working registers.
DATA REGISTER

(Output Enable 1)

(Output Enable 2)

Clock Circuit for Bus Exchange
DATA BUS

Registers are interconnected through a bus (a set of wires.) All the registers are connected to the same set of wires. At any given time, only one register can have its output enabled. It will dictate the voltages on all the bus wires. Any number of registers can read the data on the bus.

3 step data transfer process: (data from register A to B)

- 1st Clock Tick: Assert output-enable (OE) on A. After setup time, data is stable (ready) on the bus.
- 2nd Clock Tick: Assert input-enable/load (L) on B. Data on the bus is latched by B.
- 3rd Clock Tick: Take OE of A low and input-enable (L) of B low.

---

Jeff Shelton – 23 March 2015
DATA BUS

DATA TRANSFER

With proper timing of control signal changes relative to clock edges, the transfer can be accomplished in one clock period.

- Assert both OE (reg. A) and LOAD (reg. B) at the same time.
- Timing must be such that the output (data on the bus) is stable before being latched.

Additional modules (logic units) attached to the bus provide the needed processing capability.
MICROPROCESSOR REGISTERS

Processor, in addition to attaching to the main bus, it can also be directly attached to several registers.

- Operates on the output of one or more registers.
- Result can be latched to other registers.

Typical processor data transfer structure (clock is implied):

![Diagram of microprocessor registers and data bus connections](image-url)
INPUT/OUTPUT BUS

Input/Output (I/O) bus is for external (outside the processor) connections (or connections to external devices.) All information is carried on the same bus; no separation of control and address data.

The I/O bus carries: data, address, and control signals.

- Many configurations are available:
  - Dedicated control bits, address bits and data bits.
  - Dedicated control bits, shared address/data bits.
  - Shared control/address/data bits – sequential interpretation.

Devices can be added or removed easily. Boards can even be plugged-in or unplugged while the system is running – *hot swapping.*
I/O BUS INTERCHANGE

Typical sequence
- Put address on bus - alerts target device
- Assert control signal indicating intended operation (read/write).
- Put data on bus to complete interaction.

Bus protocols may be synchronous or asynchronous:
- Synchronous – CLOCK occupies one bit on the bus and all instructions must be completed within one clock cycle.
- Asynchronous – No clock, uses handshake signals to indicate when operations are complete. Handshake slows the interchange but allows more flexibility.

Bus Mastering
- **Single-Master Bus** – One unit is the bus master, other unit can access the bus only if the master voluntarily “steps down,” e.g. direct memory access (DMA).
- **Multi-Master Bus** – Several devices can arbitrarily access the bus. Additional circuitry needs to be added to adjudicate conflicts. This type of bus is the basis for multi-processor computers.
INCREASING THROUGHPUT

Two ways to increase bus performance (data transfer rate):

- Increase clock speed (make the flow faster)
- Increase number of data bits (make the pipe bigger)

Clock speed increase can be expensive – all associated components must be upgraded to run with faster clock.
PERIPHERAL INTERFACING

Generally categorized into parallel or serial methods:

- **Parallel:**
  - Multiple bits sent simultaneously across multiple lines.
  - Faster, but more costly, than serial.
  - Standards: Centronics, IEEE-488 (HPIB, GPIB)

- **Serial:**
  - One bit at a time, sent sequentially.
  - Requires fewer communication lines than parallel.
  - Standards: RS232, USB, I²C, SPI, FireWire, SATA, PCI-E
CONNECTORS

D-Sub (DB) Connectors

DB-9 (DE-9)
Serial port (RS-232).

High-density DB-15 (DE-15)
VGA port.

DB-15 (DA-15)
Game port on PC, Thick Ethernet.

DB-25
Parallel port on PC, Serial port (RS-232).

DB-37 (DC-37)
RS-423, 442, 449.

DB-50 (DD-50)
Earlier SCSI devices.

Jeff Shelton – 23 March 2015
PARALLEL COMMUNICATION

IEEE-1284 (Centronics)

- First parallel interface for printers, developed in 1970.
- Allows for transferring 8 bits at a time.
- Remains a simple way to interface an external circuit board to a computer, although parallel ports are increasingly hard to find on modern computers.

IEEE-488 (HPIB or GPIB)

- A byte-wide universal instrument interface with data rate of 1 Mbyte/sec.
- 15 instruments can be linked to a single bus cable for up to 20 m.
- Any connected device can be a “talker” (source of data) and the remaining device will be the “listener” (recipient of data). A “controller” (dictator) tells everybody what to do.
RS-232 SERIAL STANDARD


- Designed for connecting DTEs (Data Terminal Equipment) to DCEs (Data Communication Equipment). A terminal always looks like a DTE and a modem always looks like a DCE. PCs looks like a DTE with male connectors, although almost all large computers are DCE.

- A null modem (a cable that crisscrosses pins 2 and 3) is needed to connect two similar devices.

Image: https://secure.flickr.com/photos/twylo/2592395272/
RS-232 SERIAL STANDARD

Serial Communication

- Asynchronous ("handshaking") bit transmission protocol with fixed bit rate.
- Start and stop codes alert receiver of data transmission.
- Most common configuration uses eight data bits, one stop bit, and no parity bit.
RS-232 SERIAL STANDARD

Null Modem Connector (cable)

- Connecting two PCs through the serial port requires a “null modem” connector or cable.
UNIVERSAL SERIAL BUS (USB)

External expansion bus for PCs that supports plug-n-play functionality.

History
- 1995 – USB 1.0
- 2000 – USB 2.0 (480 Mbps)
- 2008 – USB 3.0 (4.8 Gbps)

Standard differential serial communication
- Slow speed: 1.5 Mbps (Low power consumption; mice and keyboards)
- Full speed: 12 Mbps (All USB hubs accommodate this speed)
- High speed: 480 Mbps (High speed data transfer, USB 2.0 or below)
- Super speed: 4.8 Gbps (Ultra-high speed data transfer, USB 3.0 or below)

Master (PC, Host Controller) – Slave (Device) type protocol

Packet based data transfer – Token + Data + Handshake packets
UNIVERSAL SERIAL BUS (USB)

Non Twisted Power Pair:
- Red: $V_{BUS}$
- Black: GND

Polyvinyl Chloride (PVC) Jacket

Outer Shield: ≥65% Interwoven Tinned Copper Braid

Inner Shield: Aluminum Metallized Polyester

Twisted Signal Pair:
- White: Data-
- Green: Data+

28 AWG Tinned Copper Drain Wire

Additional lines present in USB 3!
INTER-IC BUS (I²C)

Serial bus developed by Phillips (now NXP) Semiconductor for communicating with low-speed devices. Requires only two bidirectional communication lines, Serial Data (SDA) and Serial Clock (SCL), both of which are open-drain.

Common bus frequencies are 10 kbits/sec and 100 kbits/sec, although rates up to 3.4 Mbits/sec are possible.

Use with the Arduino is supported via the "Wire" library.
INTER-IC BUS (I²C)

Vcc  I²C supply voltage, typically ranging from 1.2 V to 5.5 V
GND  Common ground
SDA  Serial data (I²C data line)
SCL  Serial clock (I²C clock line)
Rp   Pull-up resistance (a.k.a. I²C termination)
Rs   Serial resistance
Cp   Wire capacitance
Cc   Cross channel capacitance
SERIAL PERIPHERAL INTERFACE BUS (SPI)

Synchronous serial interface standard that uses four wires. Use of four lines allows for higher throughput than I²C. Slave devices do not need a unique address, unlike I²C or GPIB. Also known as Synchronous Serial Interface (SSI).

Four interface signals are:

- SCLK: Serial Clock (from master)
- MOSI: Master Output, Slave Input
- MISO: Master Input, Slave Output
- SS: Slave Select (active low)

Arduino developers can use the SPI Library.
SERIAL PERIPHERAL INTERFACE BUS (SPI)
SERIAL COMMUNICATION

Serial ATA (SATA)
- Created in 2003 to replace ATA or Parallel-ATA
- SATA/150 – 1.5 GHz clock with 150 MB/s (1.2 gbps)
- SATA/300 (rev 2) – 3 Ghz clock with 300 MB/s (3 gbps)
  - NVIDIA nForce4 chipset in 2004
  - Also called Serial-ATA II (SATA II)
- SATA/600 (rev 3) – 600 MB/s (6 gpbs)
- 7 pin connection (3 GNDs, 2 pairs of TX/RX)
- Separate power connection

eSATA (External SATA)
- Standards published in 2004
- 2 meter cable length
LOCAL AREA NETWORK

Ethernet (IEEE 802.3)

- 10 Mbps, 100 Mbps, 1 Gbps, 10 Gbps (2008)
- Data sent in “packets” up to 1 Kbyte+, with a preamble and error checking.
- Sending protocol:
  - Wait until no activity on network.
  - Begin sending data (message) packet.
  - While sending, check simultaneously for interference (collision).
  - As long as all is clear, continue to send packets.
  - If interference is detected, jam the network intentionally (so that everybody will detect collision), then wait for a random period, and try again.
Arduino Ethernet shield R3 with micro SD connector - Assembled -

ID: 201

$45.00

<table>
<thead>
<tr>
<th>Qty.</th>
<th>Discount</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-9</td>
<td>$45.00</td>
</tr>
<tr>
<td>10-99</td>
<td>$40.50</td>
</tr>
<tr>
<td>100+</td>
<td>$36.00</td>
</tr>
</tbody>
</table>

Please enter your details below and we will send you an e-mail when this item is back in stock! You will only be e-mailed about this product!

Your Name:

Your E-mail:

NOTIFY ME WHEN IN STOCK

Image: https://www.adafruit.com/products/201
WIRELESS LAN (IEEE 802.11)

Operating frequency
- 2.4 – 2.4835 GHz (interferes with 2.4GHz cordless phone)

IEEE 802.11 (1997) – 1 and 2 Mbps raw data rate
IEEE 802.11b,e (1999, 2002) – 5.5 and 11 Mbps raw data rate (5.9 Mbps throughput)
  - Distance limited ~ 100 ft
IEEE 802.11g (2003) – 54 Mbps raw data rate
  - IEEE 802.11a (1999) – 54 Mbps raw data rate (~25 Mbps throughput)
  - Operating frequency: 5 GHz
IEEE 802.11n (2009)
  - 40 MHz channels
  - 150 Mbps throughput
**Arduino Wi-Fi Shield**
**DEV-11287**

**Description:** The Arduino WiFi Shield allows an Arduino board to connect to the internet using the 802.11 wireless specification (WiFi). It is based on the HDG104 Wireless LAN 802.11b/g System in-Package. An Atmega 32UC3 provides a network (IP) stack capable of both TCP and UDP. Use the WiFi library to write sketches which connect to the internet using the shield.

The WiFi shield connects to an Arduino board using long wire-wrap headers which extend through the shield. This keeps the pin layout intact and allows another shield to be stacked on top.

There is an onboard micro-SD card slot, which can be used to store files for serving over the network. It is compatible with the Arduino Uno and Mega. The onboard microSD card reader is accessible through the SD Library. When working with this library, SS is on Pin 4.

Arduino communicates with both the Wifi shield’s processor and SD card using the SPI bus (through the ICSP header). This is on

Image: https://www.sparkfun.com/products/11287
WIRELESS LAN (WI-FI)

Meet the Spark Core.

A tiny Wi-Fi development board that makes it easy to create internet-connected hardware. The Core is all you need to get started; power it over USB and in minutes you'll be controlling LEDs, switches and motors and collecting data from sensors over the internet!

**Wireless programming**
No need to ever plug the Core into your computer; you can write code in our web IDE and download it wirelessly to the Core.

**Arduino compatible**
Familiar with Arduino? The Core uses Wiring, the same programming language that Arduino uses. Plus, with an accessory called the Shield, you can connect the Core to any standard Arduino shield.

**Easy set-up**
How do you connect a Wi-Fi device to your network when it doesn't have a screen or keyboard? With Smart Config, the Core's one step set-up process. Bring the Core online in seconds from any iOS or Android device.

Unit price is $39

Image: https://www.spark.io/
BLUETOOTH

Named after medieval King of Denmark, Harald "Bluetooth" Gormsson

Proprietary open wireless short distance protocol (using short length radio waves), creating personal area networks (PANs)

Originally conceived as a wireless alternative to RS-232

Connecting several devices can achieve device synchronization

79 bands of 1 MHz width in the range 2402-2480 MHz
  ▪ Potential interference with 802.11

1 Mbps (ver 1.2) and 3 Mbps (ver 2.0+EDR)

Range: 1 (class 3) to 100 (class 1) meters
BLUETOOTH

Bluetooth Low Energy (BLE) Shield for Arduino 2.0

Our Price: $29.99

Availability: Usually Ships in 1 to 2 Business Days
Product Code: MKRBL1

Image: http://www.makershed.com/Bluetooth_Low_Energy_BLE_Shield_for_Arduino_2_0_p/mkrbl1.htm

JY-MCU Bluetooth Module
ZIGBEE

A simple, slow, low-power, low-cost cousin of Bluetooth; based on IEEE 802.15.4

Appropriate for certain consumer electronics and home-automation applications, as well as short-range industrial telemetry and remote control

Nodes can create an "ad-hoc" mesh network; any node may be put in charge

Uses the 2.4-GHz band

Basic data rate of 250 Kbps, but slower 28 Kbps rate extends range and reliability

Transmission distances from 10 to 1600 meters
**ZIGBEE**

**Description:** This is the XBee XB24-Z7WIT-004 module from Digi. Series 2 improves on the power output and data protocol. Series 2 modules allow you to create complex mesh networks based on the XBee ZB ZigBee mesh firmware. These modules allow a very reliable and simple communication between microcontrollers, computers, systems, really anything with a serial port! Point to point and multi-point networks are supported.

These are essentially the same hardware as the older Series 2.5, but have updated firmware. They will work with Series 2.5 modules if you update the firmware through X-CTU.

Not sure which XBee module or accessory is right for you? Check out our **XBee Buying Guide**!

**Note:** If you are looking for a simple point-to-point configuration, you might want to try the Series 1 instead. The Series 2 requires considerable setup and configuration. We highly recommend *Building Wireless Sensor Networks* as a guide for setting up mesh networks.

Image: https://www.sparkfun.com/products/10414
# Wireless Options

<table>
<thead>
<tr>
<th>Market Name</th>
<th>ZigBee®</th>
<th>---</th>
<th>Wi-Fi™</th>
<th>Bluetooth™</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard</td>
<td>802.15.4</td>
<td>GSM/GPRS CDMA1×RTT</td>
<td>802.11b</td>
<td>802.15.1</td>
</tr>
<tr>
<td>Application Focus</td>
<td>Monitoring &amp; Control</td>
<td>Wide Area Voice &amp; Data</td>
<td>Web, Email, Video</td>
<td>Cable Replacement</td>
</tr>
<tr>
<td>System Resources</td>
<td>4KB - 32KB</td>
<td>16MB+</td>
<td>1MB+</td>
<td>250KB+</td>
</tr>
<tr>
<td>Battery Life (days)</td>
<td>100 - 1,000+</td>
<td>1 - 7</td>
<td>.5 - 5</td>
<td>1 - 7</td>
</tr>
<tr>
<td>Network Size</td>
<td>Unlimited (2⁶⁴)</td>
<td>1</td>
<td>32</td>
<td>7</td>
</tr>
<tr>
<td>Maximum Data Rate (KB/s)</td>
<td>20 - 250</td>
<td>64 - 128+</td>
<td>11,000+</td>
<td>720</td>
</tr>
<tr>
<td>Transmission Range (meters)</td>
<td>1 - 100+</td>
<td>1,000+</td>
<td>1 - 100</td>
<td>1 - 10+</td>
</tr>
<tr>
<td>Success Metrics</td>
<td>Reliability, Power, Cost</td>
<td>Reach, Quality</td>
<td>Speed, Flexibility</td>
<td>Cost, Convenience</td>
</tr>
</tbody>
</table>

Table: http://www.zigbee.org/About/FAQ.aspx
NEAR FIELD COMMUNICATIONS (NFC)

Allows one-way communication between a "tag" and "device," or two-way communications between devices

"Tags" may be credit cards, key fobs, or stickers; batteries not required

Based on radio-frequency identification (RFID) technology; uses magnetic induction between two loop antennas

Communication distance of 10 cm or less

Operates at 13.56 MHz, with data rates 106, 212 or 424 Kbps

Slower than Bluetooth, but consumes much less power and doesn't require advance pairing
NEAR FIELD COMMUNICATIONS (NFC)

Adafruit PN532 NFC/RFID Controller Shield for Arduino + Extras -
ID: 789
Manufactured by: Adafruit

$39.95

<table>
<thead>
<tr>
<th>Qty.</th>
<th>Discount</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-9</td>
<td>$39.95</td>
</tr>
<tr>
<td>10-99</td>
<td>$35.96</td>
</tr>
<tr>
<td>100+</td>
<td>$31.96</td>
</tr>
</tbody>
</table>

Add to Cart: 1
ADD TO CART
IN STOCK
ADD TO WISHLIST

Image: https://www.adafruit.com/products/789
FINAL PROJECT...

- Take pictures as you go along
- Fail early and often — so you don't fail late!
- Communicate, cooperate, and compromise
- Don't rely on last-minute miracles; they rarely occur
- The truth is your friend