

L14.1 Synchronous Reset With Enable

```
library ieee;
use ieee.std_logic_1164.all;
use work.std_arith.all;
-- needed to add an integer to std_logic_vector
entity sctr is
port (reset : in std_logic;
      enb   : in std_logic;
      clk   : in std_logic;
      count : out std_logic_vector(3 downto 0));
end sctr;
architecture behavioral of sctr is
    signal count_int : std_logic_vector(3 downto 0);
begin -- behavioral
    count <= count_int;
    -- rest of architecture is in the next slide
```

L14.2 Simulation of Synchronous Reset

```
process (clk)
begin
    if rising_edge(clk) then
        if reset = '1' then
            count_int <= x"0";
        elsif enb = '1' then
            count_int <= count_int + 1;
        else count_int <= x"2";
        end if;
    end if; end process; end behavioral;
```



L14.3 Asynchronous Reset With Enable

```
process (clk)
begin
    if rising_edge(clk) then
        if reset = '1' then
            count_int <= x"0";
        elsif enb = '1' then
            count_int <= count_int + 1;
        else count_int <= x"2";
        end if;
    end if;
end process;
end behavioral;
```

L4.4 Simulation of Asynchronous Reset

```
process (clk, reset)
begin
    if reset = '1' then
        count_int <= x"0";
    elsif rising_edge(clk) then
        if enb = '1' then
            count_int <= count_int + 1;
        else count_int <= x"2";
        end if;
    end if; end process; end behavioral;
```



L14.5 NG Asynchronous Reset With Enable

```
-- same library, entity as before
architecture behavioral of actr is
    signal count_int : std_logic_vector(3 downto 0);
begin -- behavioral
    count <= count_int;
    process (clk, reset)
    begin
        if reset = '1' then
            count_int <= x"0";
        elsif rising_edge(clk) then
            if enb = '1' then
                count_int <= count_int + 1;
            end if;
        else count_int <= x"2";
        end if;
    end process;
end behavioral;
```

L14.6 Compiling NG Asynchronous Reset

When compiling, one gets these errors:

```
Warning: 'enb should be referenced in the
sensitivity list.
Warning: 'count_int' should be referenced in the
sensitivity list.
Abort: Can't handle 's' event in final equations.
```

Here is a clue. BUT What is wrong?

```
diff asyncctr.vhd ngctr.vhd
10a11
> -- same library, entity as before
22d22
<     else count_int <= x"2";
23a24
>     else count_int <= x"2";
```

L14.7 Lab 3 Assignment

Issued - Monday, March 11, 2001
MCU Checkoff - Monday, March 18, 2001
Design Checkoff - Wednesday, March 20, 2001
Lab Checkoff - Friday, April 5, 2001
Report Due - Monday, April 8, 2001

Pitch Shifting System

Digitizes and stores audio frequency input signals.
Outputs pitch shifted version plus (optionally) original signal.

Simple (crude) Pitch Shift Mechanism

Uses two buffers, samples at fixed rate and outputs at fixed rate.
Skips or adds samples to achieve pitch shifting.

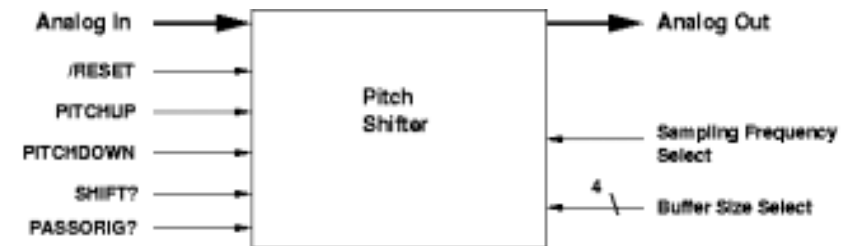
Selectable Sampling Rate (e.g., 9.6 and 19.2 kHz)

Adjustable Pitch Shift over a Wide Range

Use pushbuttons to adjust pitch shift UP and DOWN.

Selectable Buffer Size (16 Buffer Sizes)

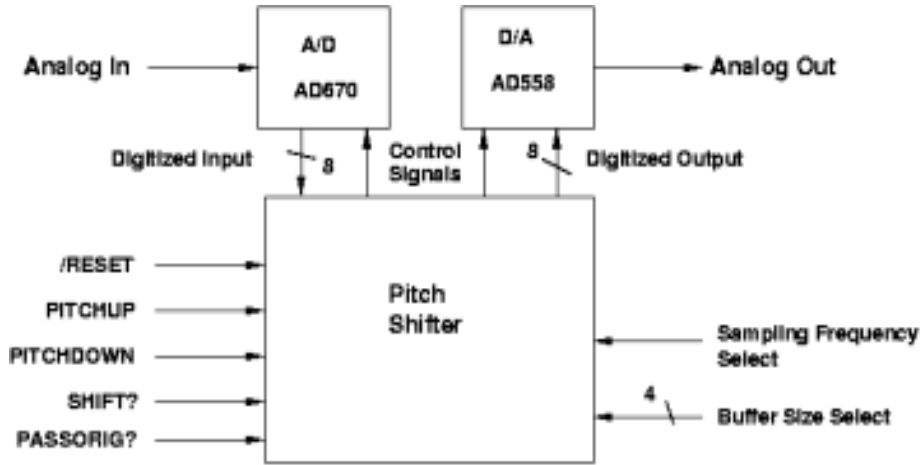
L14.8 Helium Breath



```
Analog In  Music or speech signal (or a test analog signal)
Analog Out Signal shifted up or down in pitch
/RESET     Clear pitch shift to zero
PITCHUP    Increments the amount of pitch shift
PITCHDOWN  Decrements the amount of pitch shift
SHIFT?     When asserted, output pitch shifted audio
PASSORIG?  When asserted, output original audio signal
            (When both asserted, mix)
Sampling Frequency Select - Choose one of two sampling rates.
Buffer Size Select - Choose one of 16 sampling buffer sizes.
```

L14.9 Analog-Digital Separation

As before, we (digital engineers) convert the analog input to a digital number and convert the computed digital output back to an analog voltage.



L14.10 Approach

Divide time into 'chunks'.

Write signal into a buffer (one chunk).

Use two buffers: write to one while reading from other.

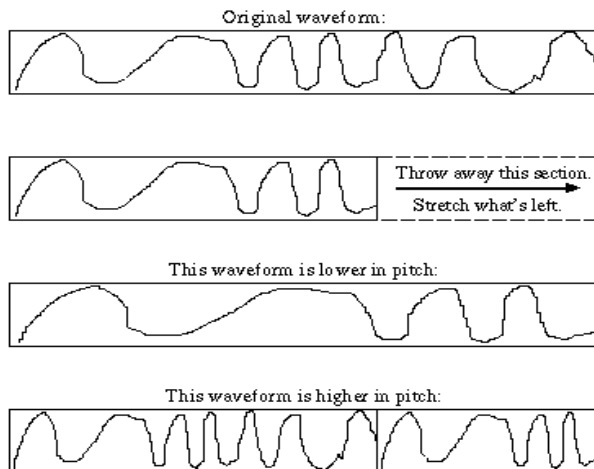
Output signal at a different rate to change pitch.

To lower pitch,
 'Stretch' signal in each chunk.
 This means we throw away end of each chunk.

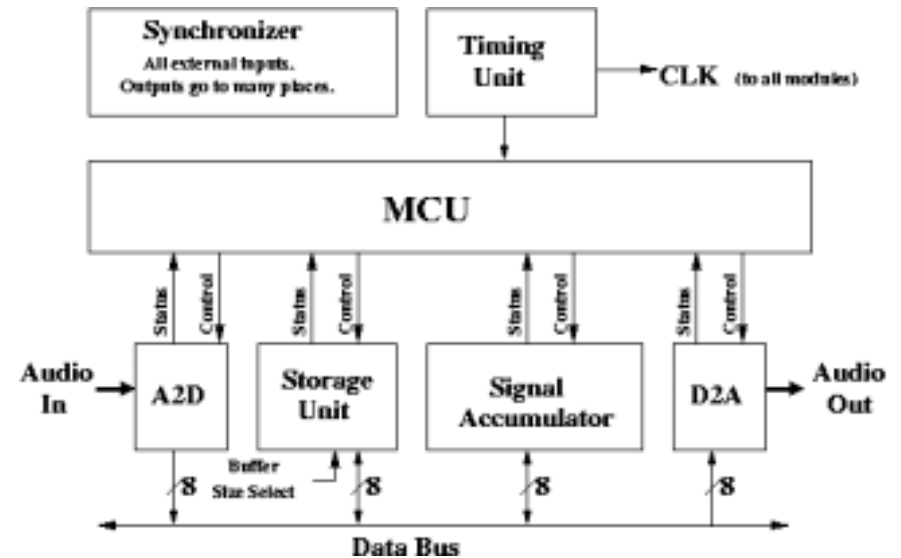
To increase pitch,
 'Squish' the signal chunk.
 Output at a higher rate.
 To fill the end of the chunk, output some part twice.

L14.11 Graphically

Stretching and Squishing



L14.12 A System Block Diagram



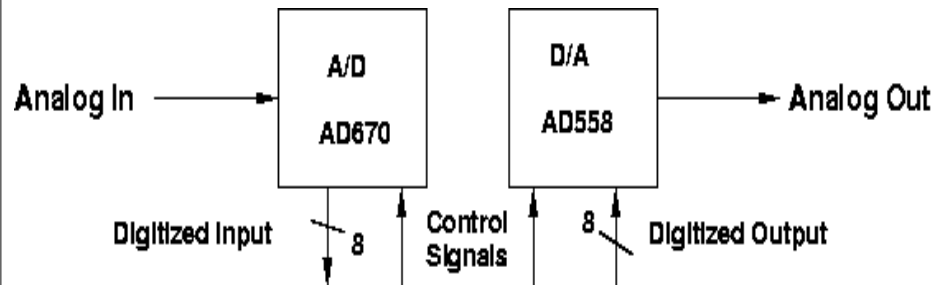
L14.13 Analog Parts

A/D: Use AD670.

D/A: Use AD558.

Data sheets are at end of lab handout.

More on these Wednesday.



L14.14 Microprogrammed Control Unit

You may choose how to build MCU. The next slide suggests a possible implementation.

A sequencer made of 2 x 74LS163.

The condition select from 75LS151 and the assertion logic done in PALs.

You should store instructions in 28F256 (flash PROM).

Since there is a 16-bit word, one uses two PROMs.

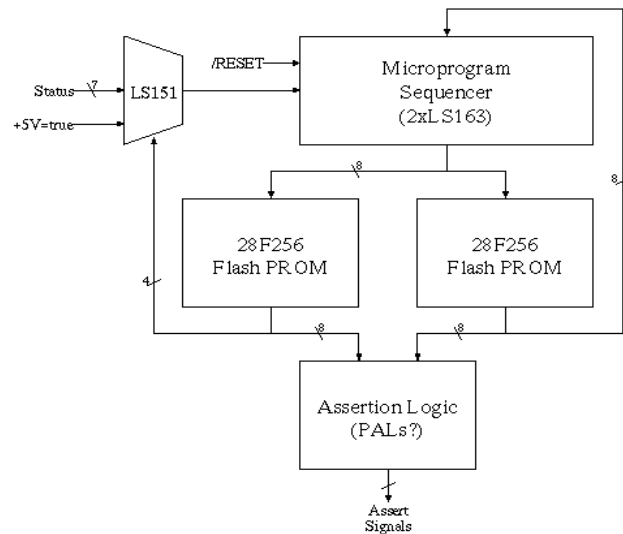
Build and test MCU and timing unit first.

A test program is provided which results in a distinctive display of blinking lights.

This ensures that at least this part is working.

	I15	I14	I13	I12	I11	I10	I09	I08	I07	I06	I05	I04	I03	I02	I01	I00
CJMP	0	C	C	C	-	-	-	-	A	A	A	A	A	A	A	A
JMP	0	1	1	1	-	-	-	-	A	A	A	A	A	A	A	A
ASSERT	1	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S

L14.15 A Possible MCU Implementation



L14.16 Timing Unit/ Signal Accumulator

Provides MCU Clock and makes sure timing constraints are observed.

Provides the sampling clock at two selectable rates.

For the MCU test it should generate yet a lower frequency – 10 to 20 Hz.

The timing unit probably should be implemented in a CPLD.

The signal accumulator is to select signals to output to the D/A.

These can be the original input signal, the pitch shifted signal, or both added together.

A possible implementation is two 74LS283s and a clearable, loadable, 8-bit register.

You may design another implementation.

L14.17 Storage Unit Block Diagram

SRAM and Memory Address Logic implements Two Storage Buffers by switching Address Counters.

The sampling counter increments by 1. 11 bits yields 2K locations.

The shifting counter is wider (17 bits => 64 x 2K) and increments by a selectable amount.

The most significant 11 bits skips the rest of the buffer or repeats starting with the beginning of the buffer.

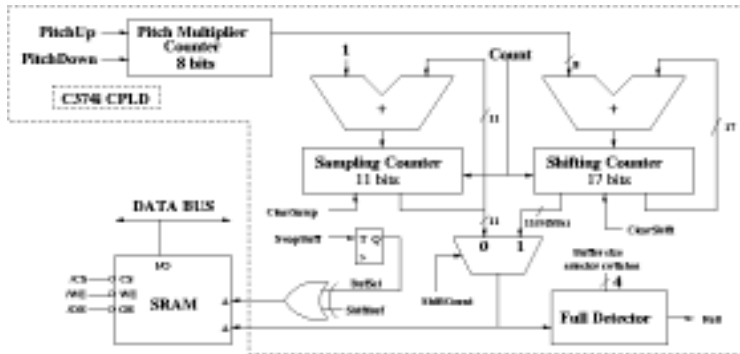
The MSB of the buffer address determines which buffer is to be used.

SwapBuff toggles the buffer. The xor of ShiftBuf and ShiftCount is used to address correct buffer.

The pitch multiplier counter provides the change for the shifting counter.

Up/Down functions controlled by pushbuttons.

The full Detector uses the size selector switch to implement variable buffer size.



L14.18 Implementation

Remember to synchronize asynchronous inputs.

Pitch Up and Down pushbuttons should produce pulses,
one per push or SLOW periodic pulses (if pushbutton held down for a while).

Do the MCU first and check it out with our program.

Debug using signal generator.

Use sine or triangle waves, NOT voice or music.

Do the signal accumulator last.

Report requirements are listed in the lab handout.