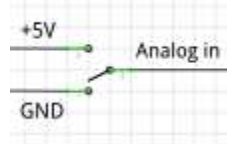


Hardware setup for Arduino Mega 2560

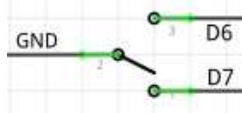
What makes the Arduino Mega 2560 perfect for this project is that it has exactly the right number of I/O pins to connect all the Altair's input/output elements:

Altair:	12	Status LEDs (INT,WO,STACK,HLTA,OUT,M1,INP,MEMR,PROT,INTE,WAIT,HLDA)
	8	Data bus LEDs (D0-D7)
	16	Address bus LEDs (A0-A15)
	16	Input switches (SW0-15)
	16	Function switches (RUN,STOP,EXAMINE,EXAMINE NEXT,DEPOSIT,DEPOSIT NEXT,RESET,CLR,PROTECT,UNPROTECT,AUX1 UP/DOWN,AUX2 UP/DOWN)
	2	Connections for serial RX/TX
	70	(digital) inputs/outputs required
Arduino Mega 2560:	54	Digital I/O pins
	16	Analog input pins (can be used for digital input)
	70	I/O lines available

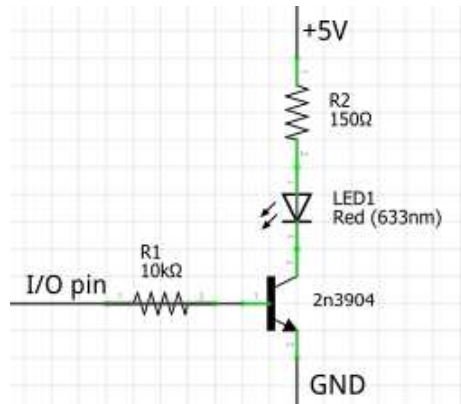
For simplicity, we connect the Altair's 16 input switches (SW0-15) to the Arduino's 16 analog inputs. These are SPDT On-On switches that alternate between two settings:



For the function switches we use the Arduino's internal pull-up resistors and just switch GND to the appropriate digital input. These are momentary SPDT (On)-Off-(On) switches. For example, the EXAMINE/EXAMINE NEXT switch is connected like this:



To drive the output LEDs we just use a simple transistor switch circuit for each LED (to prevent overload on the Arduino if too many of the LEDs are on at the same time):



See the following page for a mapping of exactly which front panel element maps to which Arduino I/O pin.

Pin Mapping for Arduino Mega 2560

Mapping Arduino pin to Altair element

Arduino Pin	Altair Element	Direction	Arduino Pin	Altair Element	Direction
D0	Serial RX	in	D35	A10	out
D1	Serial TX	out	D36	A9	out
D2	AUX2 UP	in	D37	A8	out
D3	AUX2 DOWN	in	D38	INTE	out
D4	STEP	in	D39	PROT	out
D5	SLOW	in	D40	WAIT	out
D6	EXAMINE	in	D41	HLDA	out
D7	EXAMINE NEXT	in	D42	D7	out
D8	DEPOSIT	in	D43	D6	out
D9	DEPOSIT NEXT	in	D44	D5	out
D10	OUT	out	D45	D4	out
D11	M1	out	D46	D3	out
D12	INP	out	D47	D2	out
D13	MEMR	out	D48	D1	out
D14	AUX1 UP	in	D49	D0	out
D15	AUX1 DOWN	in	D50	HLTA	out
D16	PROTECT	in	D51	STACK	out
D17	UNPROTECT	in	D52	WO	out
D18	RESET	in	D53	INT	out
D19	CLR	in	A0	SW0	in
D20	RUN	in	A1	SW1	in
D21	STOP	in	A2	SW2	in
D22	A0	out	A3	SW3	in
D23	A1	out	A4	SW4	in
D24	A2	out	A5	SW5	in
D25	A3	out	A6	SW6	in
D26	A4	out	A7	SW7	In
D27	A5	out	A8	SW8	In
D28	A6	out	A9	SW9	In
D29	A7	out	A10	SW10	In
D30	A15	out	A11	SW11	In
D31	A14	out	A12	SW12	In
D32	A13	out	A13	SW13	In
D33	A12	out	A14	SW14	In
D34	A11	out	A15	SW15	In

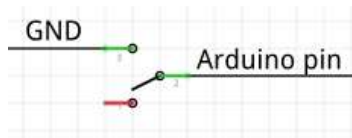
Mapping Altair element to Arduino pin

Altair Element	Arduino Pin	Direction	Altair Element	Arduino Pin	Direction
SW0	A0	in	A0	D22	out
SW1	A1	in	A1	D23	out
SW2	A2	in	A2	D24	out
SW3	A3	in	A3	D25	out
SW4	A4	in	A4	D26	out
SW5	A5	in	A5	D27	out
SW6	A6	in	A6	D28	out
SW7	A7	In	A7	D29	out
SW8	A8	in	A8	D37	out
SW9	A9	in	A9	D36	out
SW10	A10	in	A10	D35	out
SW11	A11	in	A11	D34	out
SW12	A12	in	A12	D33	out
SW13	A13	in	A13	D32	out
SW14	A14	in	A14	D31	out
SW15	A15	in	A15	D30	out
STOP	D21	in	D0	D49	out
RUN	D20	in	D1	D48	out
SINGLE STEP	D4	in	D2	D47	out
SLOW	D5	in	D3	D46	out
EXAMINE	D6	in	D4	D45	out
EXAMINE NEXT	D7	in	D5	D44	out
DEPOSIT	D8	in	D6	D43	out
DEPOSIT NEXT	D9	in	D7	D42	out
RESET	D18	in	INT	D53	out
CLR	D19	in	WO	D52	out
PROTECT	D16	in	STACK	D51	out
UNPROTECT	D17	in	HLTA	D50	out
AUX1 UP	D14	in	OUT	D10	out
AUX1 DOWN	D15	in	M1	D11	out
AUX2 UP	D2	in	INP	D12	out
AUX2 DOWN	D3	in	MEMR	D13	out
Serial RX	D0	in	PROT	D39	out
Serial TX	D1	out	INTE	D38	out
			WAIT	D40	out
			HLDA	D41	out

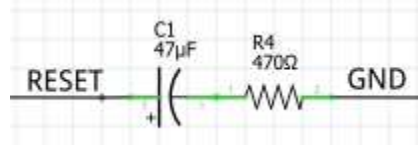
Hardware setup for Arduino Due

The setup for the Arduino Due is similar to the Arduino Mega (see above), just the pins are different (because the simulator accesses some of the pins directly by their processor register to increase efficiency). See the next page for the Arduino Due connections.

- LED drivers: Same as for the Arduino Mega. Note that the LED driver still uses +5V even though the Due runs at 3.3V (because the voltage is only used for lighting the LEDs).
- Function switches: Same as for the Arduino Mega (only different pins)
- Address switches: On the Due we can access the analog input pins just like digital inputs, including the ability to switch on an internal pullup resistor. That slightly simplifies the setup for the SW0-SW15 switches as we do not have to connect Vcc to the switches, only GND:



- The reset circuit on (some) Due boards is not very reliable at power-up (this seems to be a common problem), leaving the Due sometimes in a blocked state after power-up. If you experience this, a simple workaround is to add a capacitor and resistor to keep the RESET line low for a bit longer at power-up. This has worked fine for me and not caused any side effects.



- In the mapping on the next pages you will see some Arduino pins shown with two labels, for example A0 (D54). In this case, A0 is the label on the board and D54 is the digital pin number that relates to that connection (i.e. Analog input 0 (A0) can be accessed as Digital input 54 (D54))

Pin Mapping for Arduino Due

Mapping Arduino pin to Altair element

Arduino Pin	Altair Element	Direction	Arduino Pin	Altair Element	Direction
D0	Unused		D36	A2	out
D1	Unused		D37	A3	out
D2	INT	out	D38	A4	out
D3	WO	out	D39	A5	out
D4	STACK	out	D40	A6	out
D5	HLTA	out	D41	A7	out
D6	OUT	out	D42	SW14	in
D7	M1	out	D43	SW15	in
D8	INP	out	D44	A15	out
D9	MEMR	out	D45	A14	out
D10	WAIT	out	D46	A13	out
D11	D7	out	D47	A12	out
D12	INTE	out	D48	A11	out
D13	PROT	out	D49	A10	out
D14	D4	out	D50	A9	out
D15	D5	out	D51	A8	out
D16	SW9	in	D52	RESET	in
D17	SW8	in	D53	CLR	in
D18	Serial TX	out	A0 (D54)	STEP	in
D19	Serial RX	in	A1 (D55)	SLOW	in
D20	RUN	in	A2 (D56)	EXAMINE	in
D21	STOP	in	A3 (D57)	EXAMINE NEXT	in
D22	HLDA	out	A4 (D58)	DEPOSIT	in
D23	SW10	in	A5 (D59)	DEPOSIT NEXT	in
D24	SW11	in	A6 (D60)	PROTECT	in
D25	D0	out	A7 (D61)	UNPROTECT	in
D26	D1	out	A8 (D62)	SW0	in
D27	D2	out	A9 (D63)	SW1	in
D28	D3	out	A10 (D64)	SW2	in
D29	D6	out	A11 (D65)	SW3	in
D30	AUX1 UP	in	DAC0 (D66)	SW4	in
D31	AUX1 DOWN	in	DAC1 (D67)	SW5	in
D32	AUX2 UP	in	CANRX (D68)	SW6	in
D33	AUX2 DOWN	in	CANTX (D69)	SW7	in
D34	A0	out	SDA1 (D70)	SW12	in
D35	A1	out	SCL1 (D71)	SW13	in

Mapping Altair element to Arduino pin

Altair Element	Arduino Pin	Direction	Altair Element	Arduino Pin	Direction
SW0	A8 (D62)	in	A0	D34	out
SW1	A9 (D63)	in	A1	D35	out
SW2	A10 (D64)	in	A2	D36	out
SW3	A11 (D65)	in	A3	D37	out
SW4	DAC0 (D66)	in	A4	D38	out
SW5	DAC1 (D67)	in	A5	D39	out
SW6	CANRX (D68)	in	A6	D40	out
SW7	CANTX (D69)	In	A7	D41	out
SW8	D17	in	A8	D51	out
SW9	D16	in	A9	D50	out
SW10	D23	in	A10	D49	out
SW11	D24	in	A11	D48	out
SW12	SDA1 (D70)	in	A12	D47	out
SW13	SCL1 (D71)	in	A13	D46	out
SW14	D42	in	A14	D45	out
SW15	D43	in	A15	D44	out
STOP	D21	in	D0	D25	out
RUN	D20	in	D1	D26	out
SINGLE STEP	A0 (D54)	in	D2	D27	out
SLOW	A1 (D55)	in	D3	D28	out
EXAMINE	A2 (D56)	in	D4	D14	out
EXAMINE NEXT	A3 (D57)	in	D5	D15	out
DEPOSIT	A4 (D58)	in	D6	D29	out
DEPOSIT NEXT	A5 (D59)	in	D7	D11	out
RESET	D52	in	INT	D2	out
CLR	D53	in	WO	D3	out
PROTECT	A6 (D60)	in	STACK	D4	out
UNPROTECT	A7 (D61)	in	HLTA	D5	out
AUX1 UP	D30	in	OUT	D6	out
AUX1 DOWN	D31	in	M1	D7	out
AUX2 UP	D32	in	INP	D8	out
AUX2 DOWN	D33	in	MEMR	D9	out
Serial RX	D19	in	PROT	D13	out
Serial TX	D18	out	INTE	D12	out
			WAIT	D10	out
			HLDA	D22	out

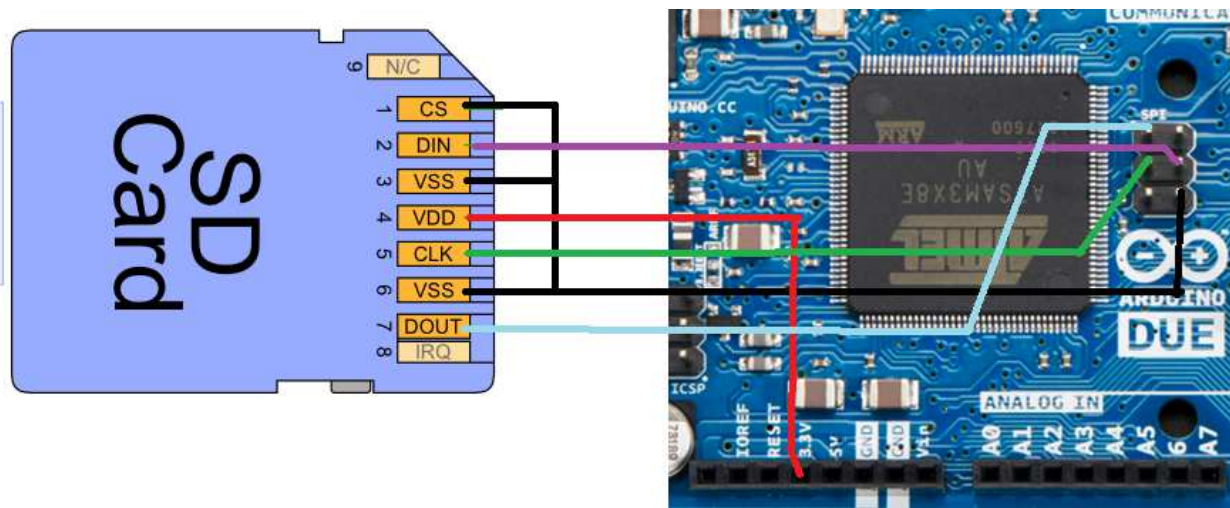
Wiring an SD card to the Arduino Due

Standard Arduino SD card shields will not work with the Arduino Due because the SPI pins are not connected to the D13-D11 pins as in other Arduino board. That is a good thing because we're already using the D13-D11 pins for front panel elements.

On the DUE, the SPI pins are **only** available on the separate 2-row, 6-pin SPI header (labeled "SPI" on the board).

There are commercial products such as the SparkFun Level Shifting microSD Breakout board that provide an SD card slot. That board is certainly works but is overkill since the Arduino DUE (like the SD cards) operates on 3.3V so no level shifting is required.

In fact, an SD card can be wired directly (without any other required electronic elements) to the SPI header on the Due:



Unfortunately, there is no 3.3V pin on the SPI header on the Arduino Due so that wire must go to the (separate) 3.3V pin. Do not wire the 5V output from the SPI header to the card. Doing so could damage the SD card and/or the Due.

The CS (chip select) pin is wired directly to GND, so the SD card is always selected. That saves us from having to find another I/O pin on the Arduino to use for chip select. The SD library on the Arduino requires to specify a Chip Select output pin but the simulator software sets that to the HLDA light output pin which as a side effect gives a "sd card active" visual indicator.

I recommend getting a microSD card with a microSD-to-SD adapter. Take the microSD card out of the adapter and create a cable by soldering wires directly to the adapter's pins and connecting them to the SPI header using the wiring given above. The adapter now serves as the socket for the microSD card, which can be plugged in and taken out easily.