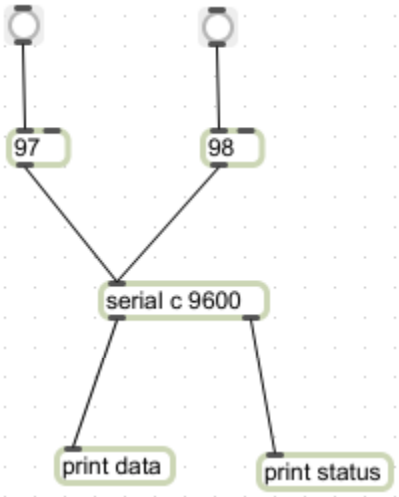
Simple Serial

MaxMSP ➔ Arduino




```
int inByte = 0;         // incoming serial byte

void setup()
{
  Serial.begin(9600);
  pinMode(13, OUTPUT);      // connect an LED here
}

void loop()
{
  inByte = Serial.read();
  if (inByte == 97) digitalWrite(13, HIGH);   // letter 'a' turns LED on
  if (inByte == 98) digitalWrite(13, LOW);    // letter 'b' turns LED off
}
```

- Serial.read receives one byte at a time
- Click on MaxMSP “97” message: Arduino pin 13 goes high
- Click on MaxMSP “98” message: Arduino pin 13 goes low
- These numbers are single ASCII byte values, arbitrarily chosen

ASCII Code

each byte value corresponds to a character

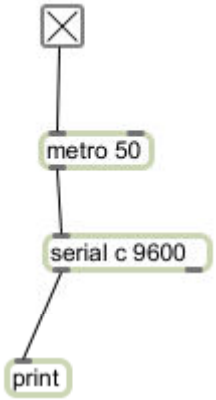
Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Source: www.LookupTables.com

Simple Serial

Arduino ➔ MaxMSP

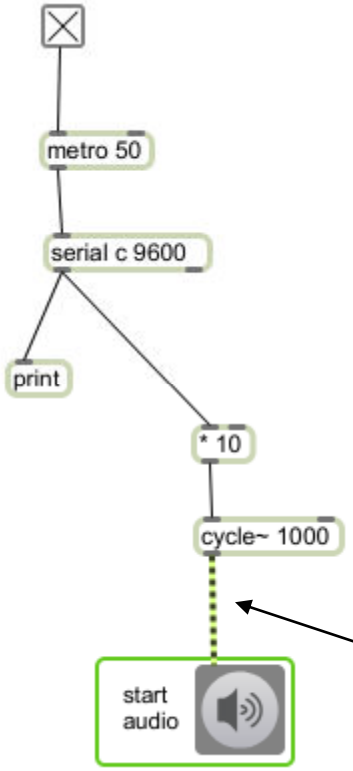
```
int sensorValue = 0;

void setup()
{
  Serial.begin(9600);
}

void loop()
{
  sensorValue = analogRead(0);
  sensorValue = map(sensorValue, 0, 1023, 0, 255);
  Serial.write(sensorValue);
  delay(100);
}
```



- Sends one byte at a time
- Only a single stream of numbers (i.e. data from one sensor) can be sent
- Values are limited to the range 0-255
- “map” command takes 0-1023 sensor range and reduces it to a range of 0-255

Simple Serial

Arduino sensor controls sound in MaxMSP

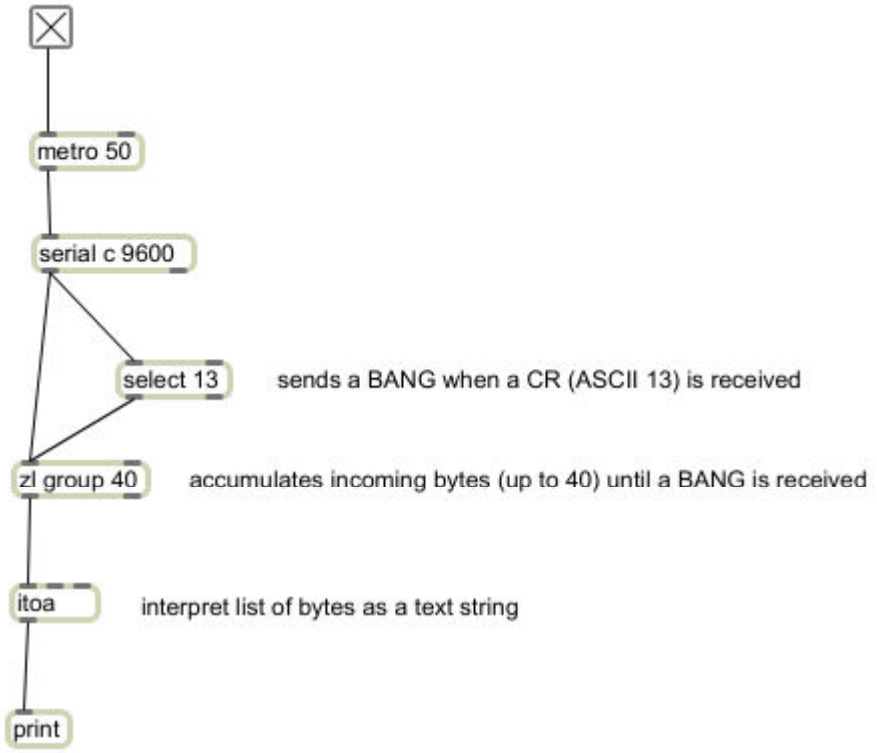


- Add these blocks to generate a tone
- MSP does realtime audio processing
- All MSP objects end in “~”
- MSP audio interconnects are dashed lines

Using Formatted Data

Arduino ➔ MaxMSP

```
int sensorValue = 0;

void setup()
{
  Serial.begin(9600);
}

void loop()
{
  sensorValue = analogRead(0);
  //sensorValue = map(sensorValue, 0, 1023, 0, 255);
  Serial.println(sensorValue);
  delay(100);
}
```

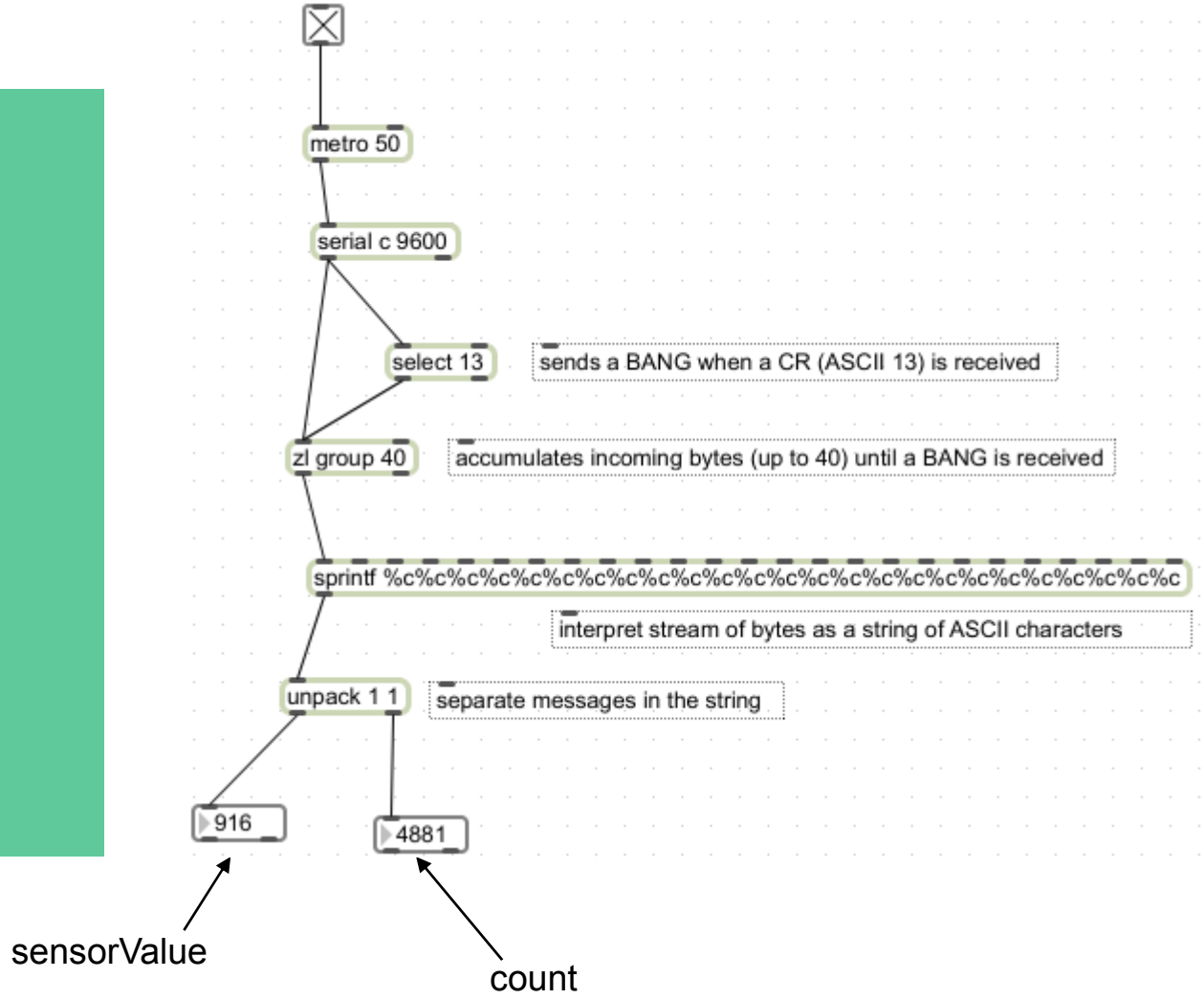


- Formatting allows multi-byte packets to be sent
- “Real world” language can be used
- Data from multiple sensors can be sent
- Numbers can take any format or range of values (no need to use “map”)
- Serial.print**ln** automatically adds ASCII 13 to the end of every packet sent

Multiple Messages

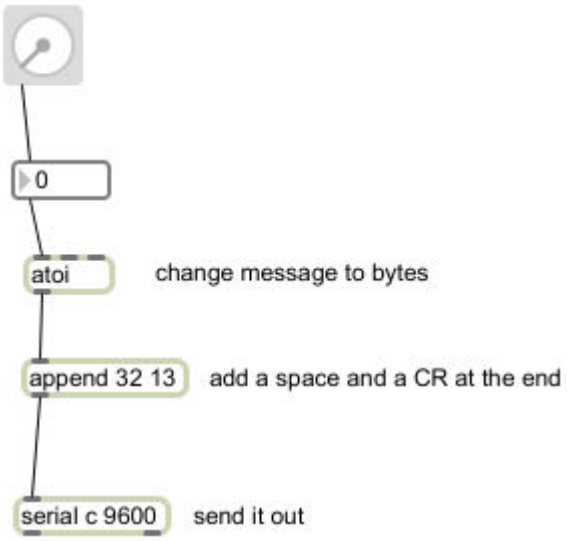
Arduino ➔ MaxMSP

```
int sensorValue = 0;
int count;

void setup()
{
  Serial.begin(9600);
}

void loop()
{
  sensorValue = analogRead(0);
  count = count + 1;
  Serial.print(sensorValue);
  Serial.write(32);
  Serial.println(count);
  delay(100);
}
```



Formatted Serial

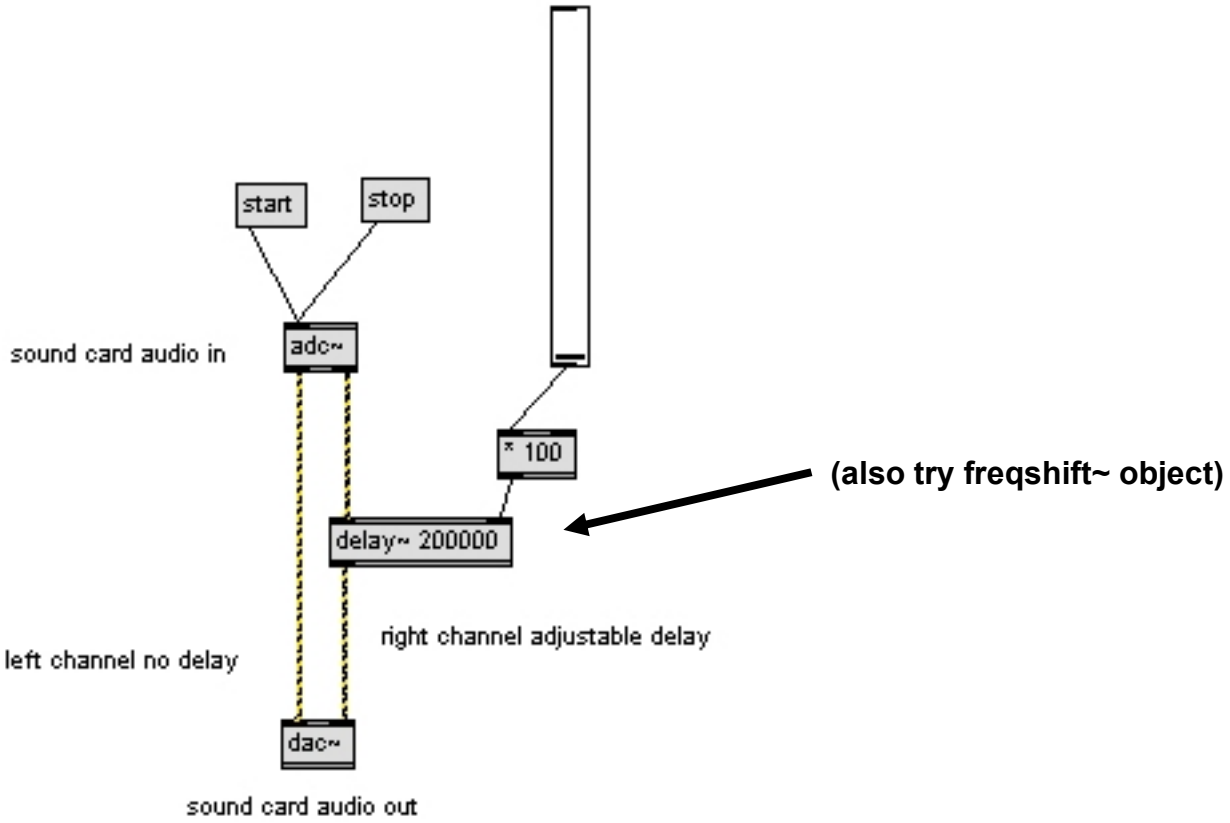
MaxMSP ➔ Arduino




```
int inByte = 0;         // incoming serial byte
char buffer[40];
int index = 0;
int value;

void setup()
{
  Serial.begin(9600);
  pinMode(11, OUTPUT);
}

void loop()
{
    index = 0;
    do
    {
       buffer[index] = Serial.read();                  // get a byte from the serial port
       if (buffer[index]!=-1)  index = index+1;        // -1 if no byte is present
    } while (buffer[index-1] != 32);
    //keep collecting bytes until a space is received

    value = atoi(buffer);       // interpret buffer string as an integer
    analogWrite(11,value);  // set brightness of an LED on pin 11
}
```

Audio Processing

External Sound Input



Assignment for Thursday 8 April 2010:

One page proposal for Final Project

This proposal is due in class on Thursday April 8th.
The project will be presented in class on May 6th

Final Project

For this project, you will connect multiple sensors (such as switches, potentiometers, flex sensors, light sensors, thermistors, ping distance sensors, IR distance sensors, accelerometers, etc…) to the Arduino. and use them to control audio and/or video in MaxMSP.

In your proposal please include a description of what you want your project to do and how it will react and interact with its environment. Make a list of specifically which sensors you intend to use and provide a breakdown of the system's intended behavior describing how the hardware and software will function in a range of situations , for example:

– ping distance sensor controls speed of video playback based upon viewer location

– video track is selected by buttons on keyboard

-- audio is pitch shifted by potentiometer setting

-- tilting the keyboard triggers new audio track…

Try to make the system as engaging/interesting as possible. It may help to first choose a category for the device you want to make such as “toy”, “game”, “sculpture”, “fashion accessory”, or “musical instrument”. It is also a good idea to decide first whether this is primarily a sound or video based system such as a musical instrument, effects box, or interactive video installation.

The primary focus over the remaining weeks of the class will be working together to make sure each of you can realize your projects as planned. The grade for this project will be based upon a combination of technical proficiency (in terms of software, hardware, and mechanical/visual design) and conceptual development.