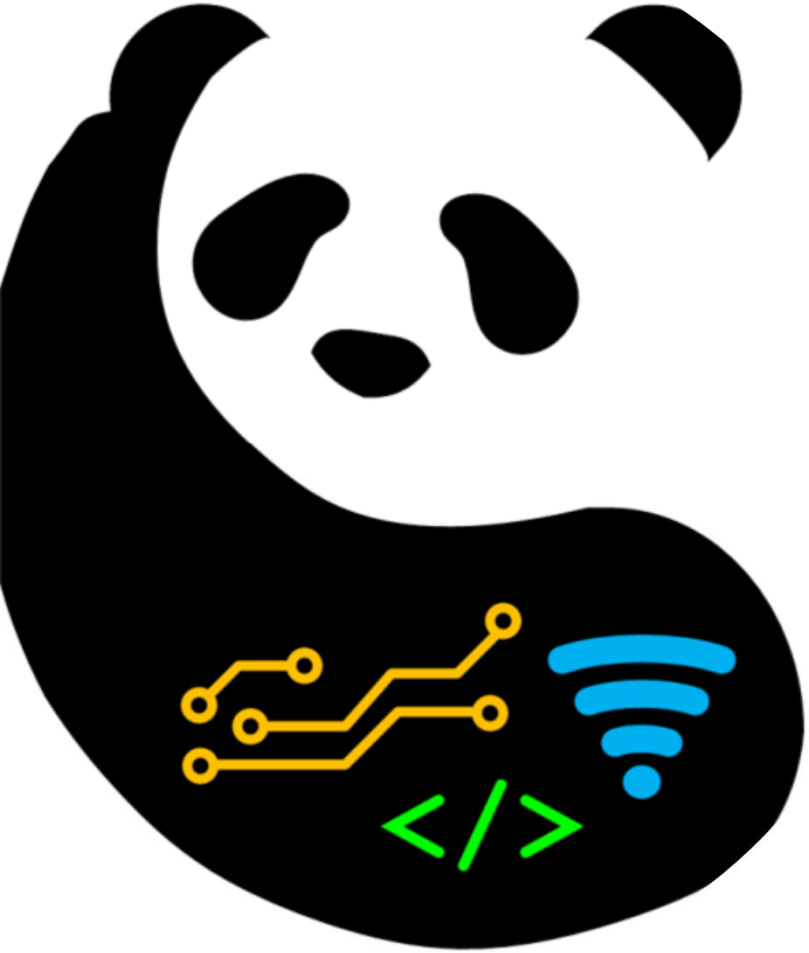
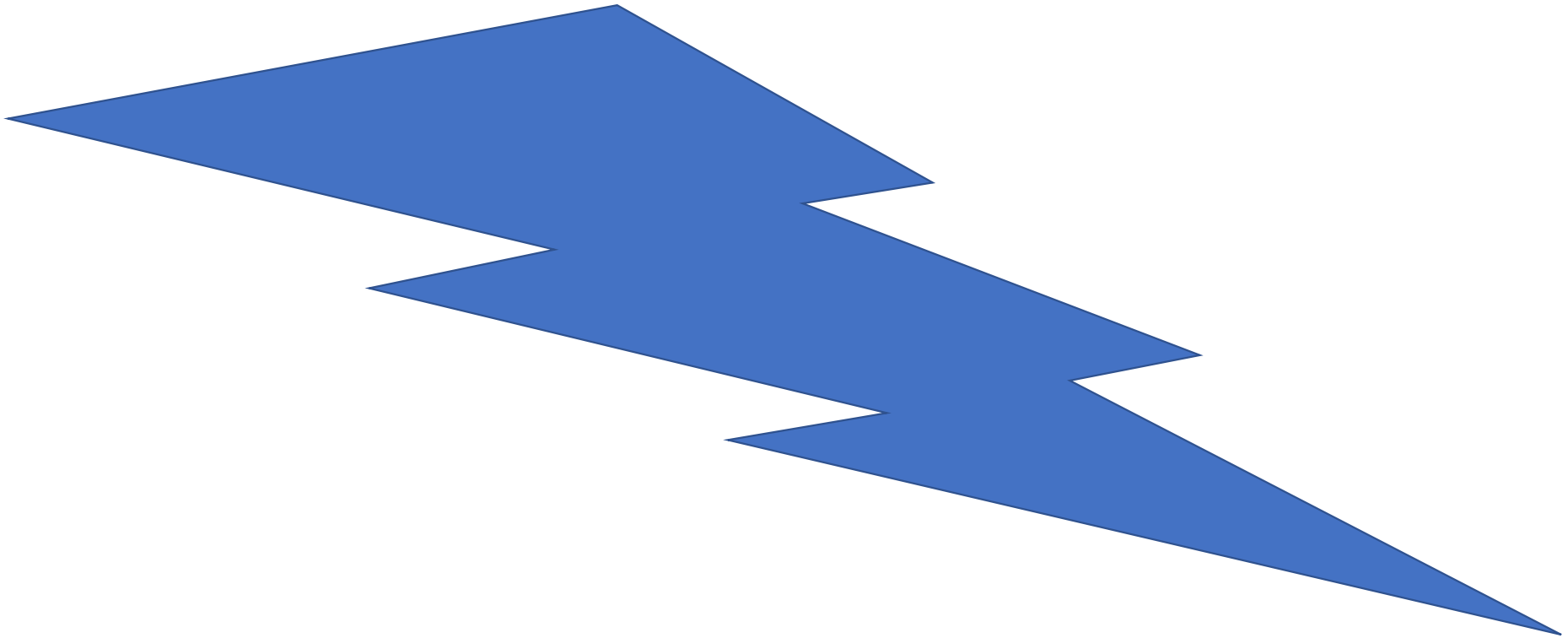
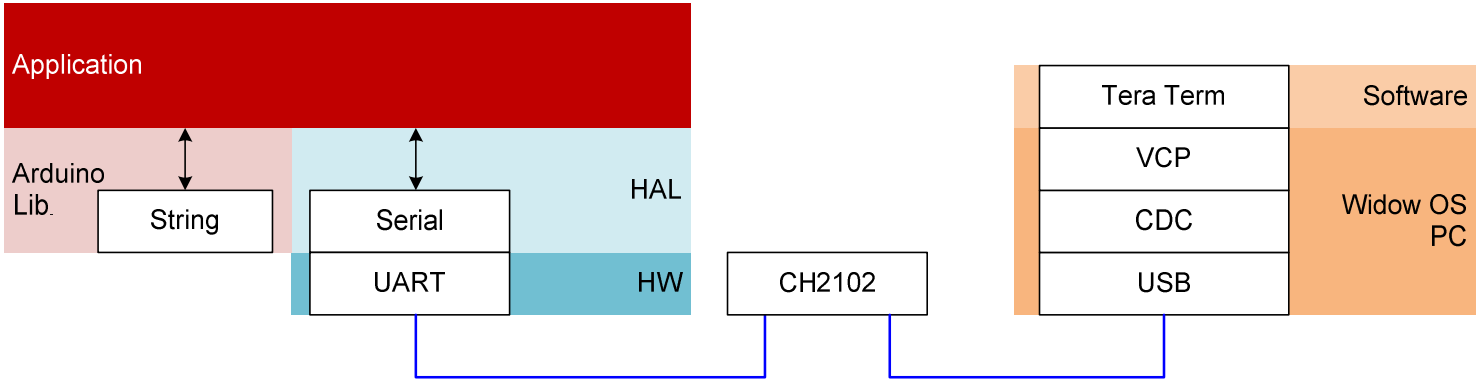


Lab. 3 User Command Interface
- UART, Serial Interface
- ASCII, Terminal Tool
- Linear buffer
- String, String Object
- String comparison
- Task synchronization
  -- Requester and Servicer
  -- Combine scheduled and unscheduled tasks

L03p01



Background knowledge



- + knowledge from Lab. 1
- + knowledge from Lab. 2

Tera Term: Setting

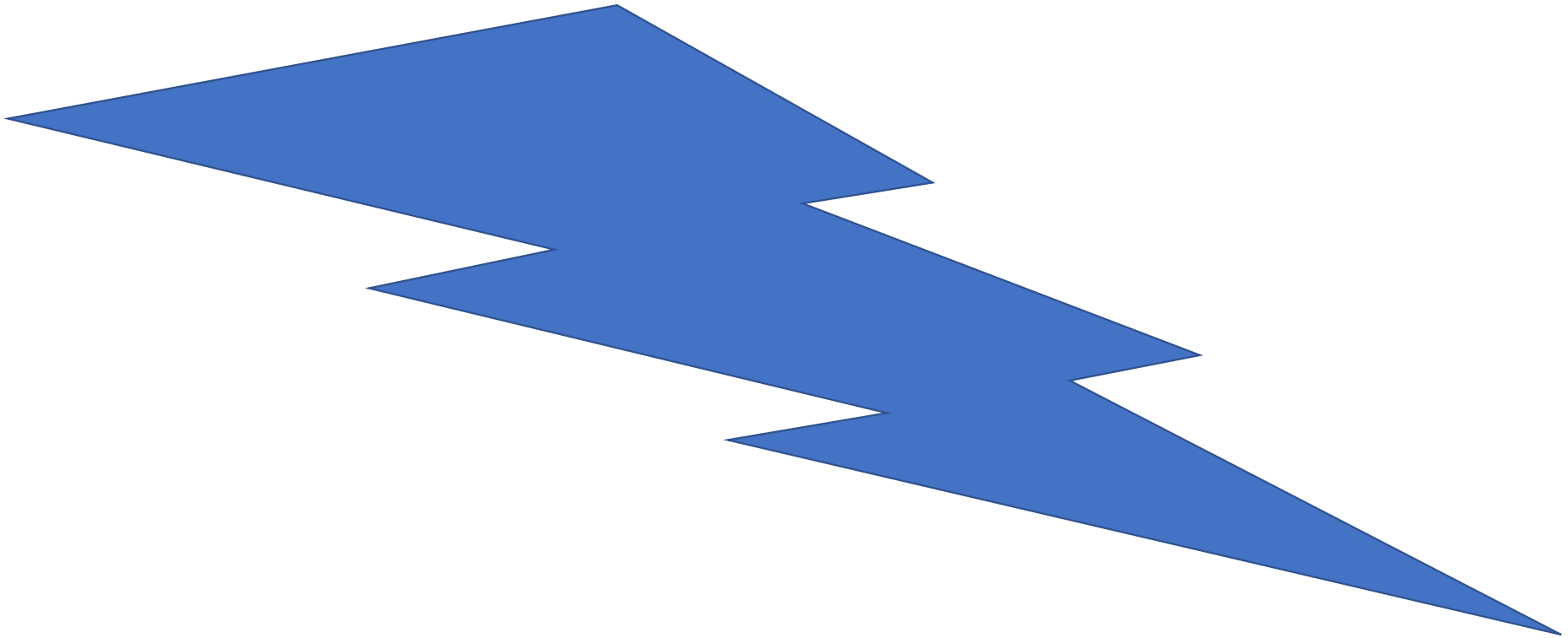
- Setup >> Serial port: 115200 bps, 8, N, 1
- Setup >> Terminal: New line LF, Local echo
- Setup >> General: Language English
- … Tera Term access the port → File >> New connection
- … Tera Term release the port → File >> Disconnection

Lab03p01: ASCII

```
14  void loop() {
15      UserCommand();
16  }
17  //#################################
18  void UserCommand() {
19      int rb;
20      if (Serial.available() > 0) {
21          rb = Serial.read();
22          Serial.print("> ");
23          Serial.print(char(rb));
24          Serial.print(" ");
25          Serial.println(rb, HEX);
26      }
27  }
```

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

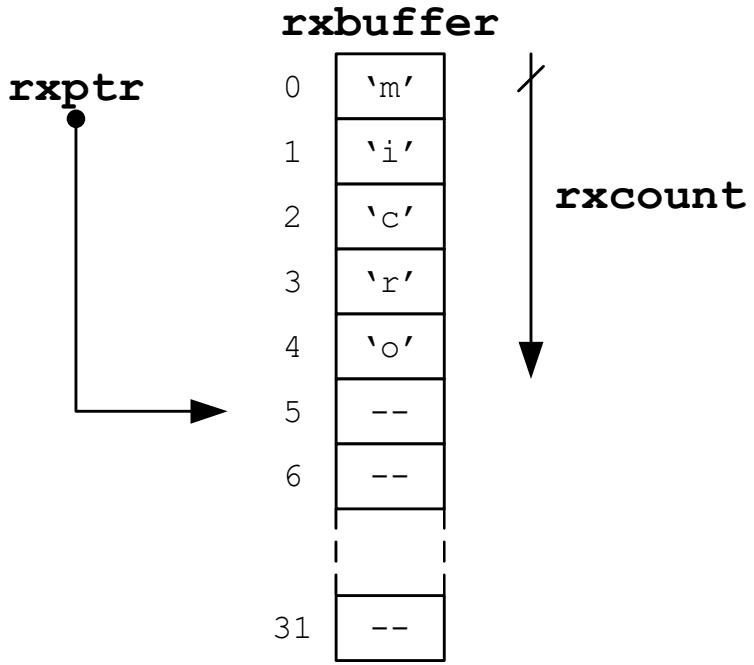
L03p02



Lab03p02: Linear buffer

```
2   char rxbuffer[32], rxcount, *rxptr;
```

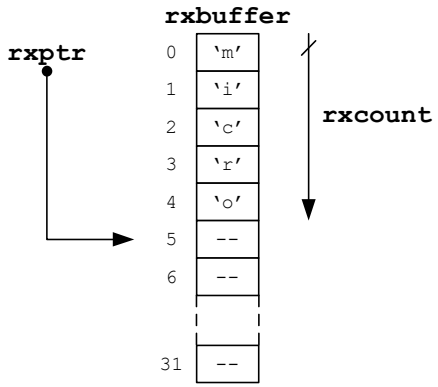
```
6   void setup() {
7       delay(3000);
8       // H/W initialize
9       Serial.begin(115200);
10      // Variables and state initialize
11      rxcount = 0;
12      rxptr = &rxbuffer[0];
13      // End of setup
14      Serial.println("\n\r\n\r\n\r#### End of setup ####");
15      Serial.print('>');
16  }
17  void loop() {
18      UserCommand();
19  }
```



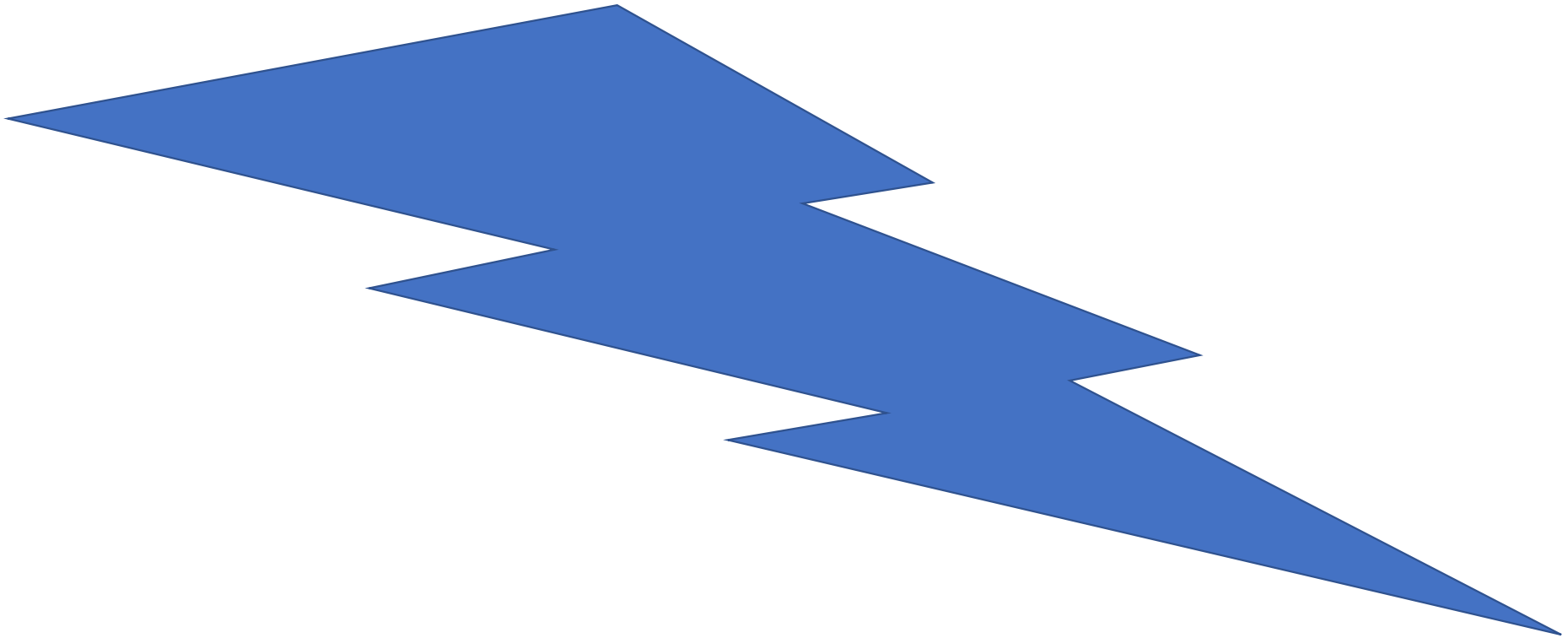
Lab03p02: Save to and Read from Linear buffer

```
30  void UserCommand() {
31      int rb;
32      while (Serial.available() > 0) {
33          rb = Serial.read();
34          if (rb == 0x0A) {
35              SendOut();
36              rxcount = 0;
37              rxptr = &rxbuffer[0];
38          }
39          else if (rb >= 0x20) {
40              if (rxcount < 32) {
41                  *rxptr++ = (char)rb;
42                  rxcount++;
43              }
44          }
45      }
46  }
```

```
21  void SendOut() {
22      Serial.print(":");
23      rxptr = &rxbuffer[0];
24      while(rxcount > 0) {
25          Serial.print(char(*rxptr++));
26          rxcount--;
27      }
28      Serial.print("\n\r>");
29  }
```



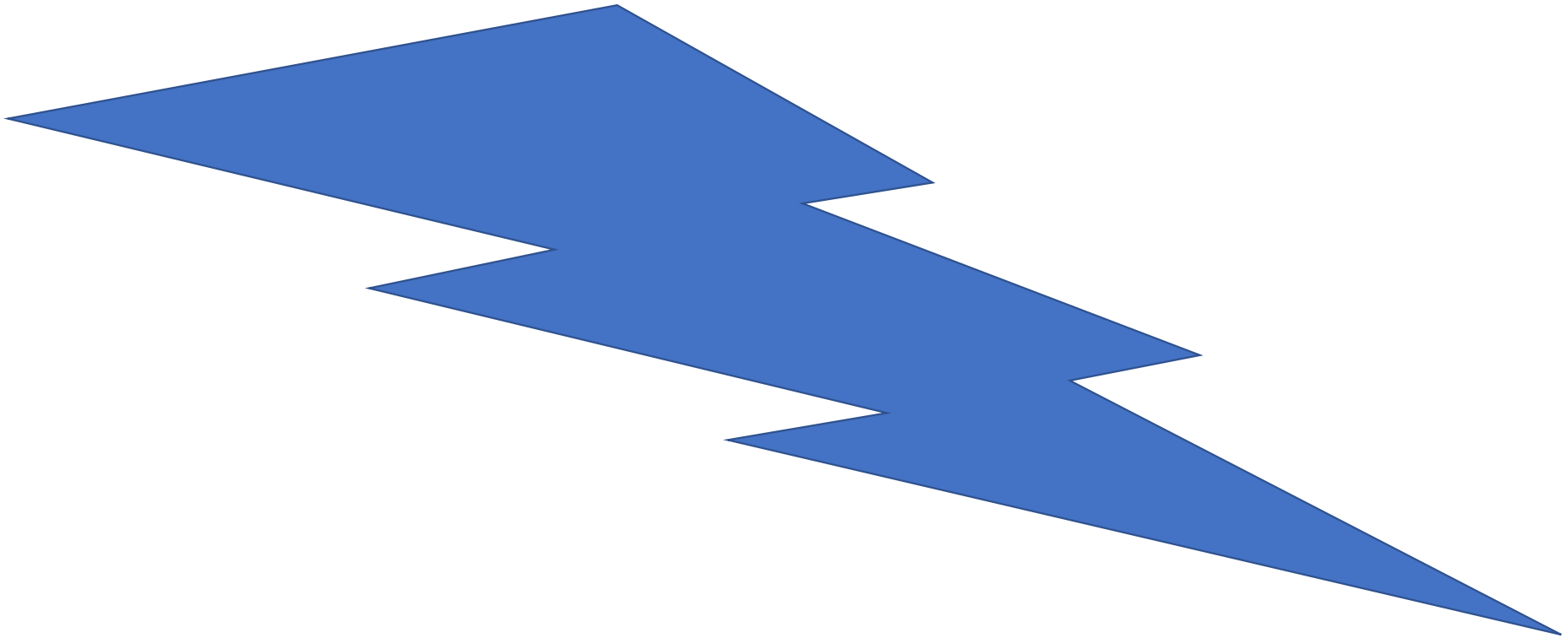
L03p03



Lab03p03: Control ASCII – 0x08 Backspace

```
30  void UserCommand() {
31      int rb;
32      while (Serial.available() > 0) {
33          rb = Serial.read();
34          if (rb == 0x0A) {
35              SendOut();
36              rxcount = 0;
37              rxptr = &rxbuffer[0];
38          }
39          else if (rb == 0x08) {
40              if (rxcount > 0) {
41                  Serial.print(' ');
42                  Serial.print(char(8));
43                  rxcount--;
44                  rxptr--;
45              }
46              else {
47                  Serial.print('>');
48              }
49          }
50          else if (rb >= 0x20) {
51              if (rxcount < 32) {
52                  *rxptr++ = (char)rb;
53                  rxcount++;
54              }
55          }
56      }
57  }
```

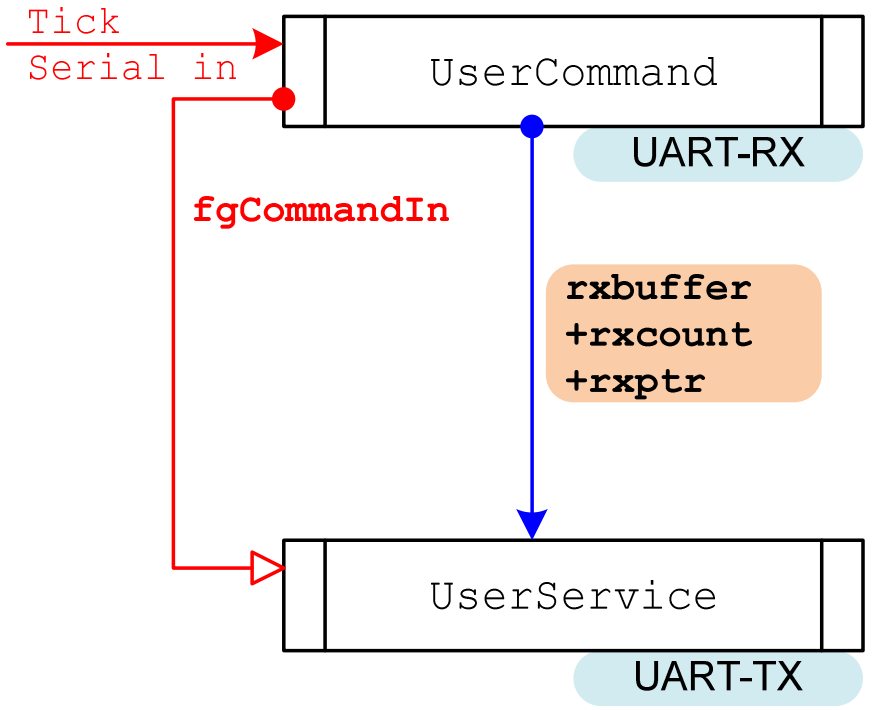
L03p04



Lab03p04: Requester and Servicer

```
19  void loop() {
20      UserCommand();
21      if (fgCommandIn == 1) UserService();
22  }
```

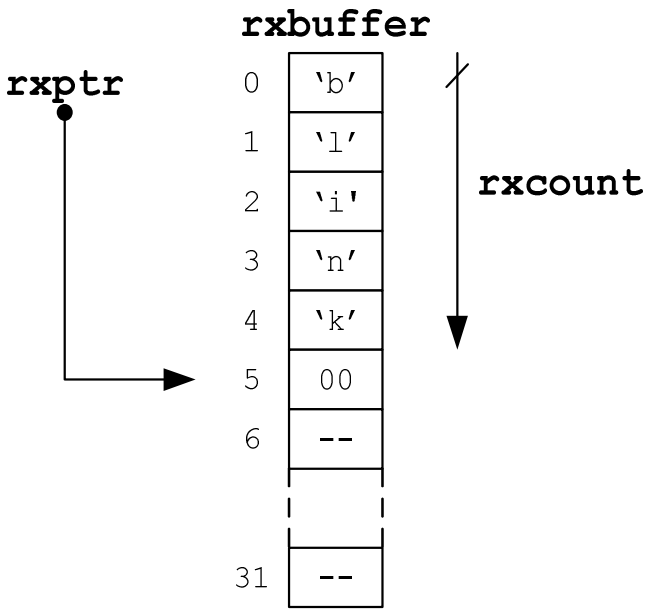
```
24  void UserCommand() {
25      int rb;
26      while (Serial.available() > 0) {
27          rb = Serial.read();
28          if (fgCommandIn == 1) return;
29          if (rb == 0x0A) {
30              *rxptr = 0;
31              rxcount = 0;
32              rxptr = &rxbuffer[0];
33              fgCommandIn = 1;
34          }
35          else if (rb == 0x08) {
36              if (rxcount > 0) {
37                  Serial.print(' ');
38                  Serial.print(char(8));
39                  rxcount--;
40                  rxptr--;
41              }
42              else {
43                  Serial.print('>');
44              }
45          }
46          else if (rb >= 0x20) {
47              if (rxcount < 32) {
48                  *rxptr++ = (char)rb;
49                  rxcount++;
50              }
51          }
52      }
53  }
```



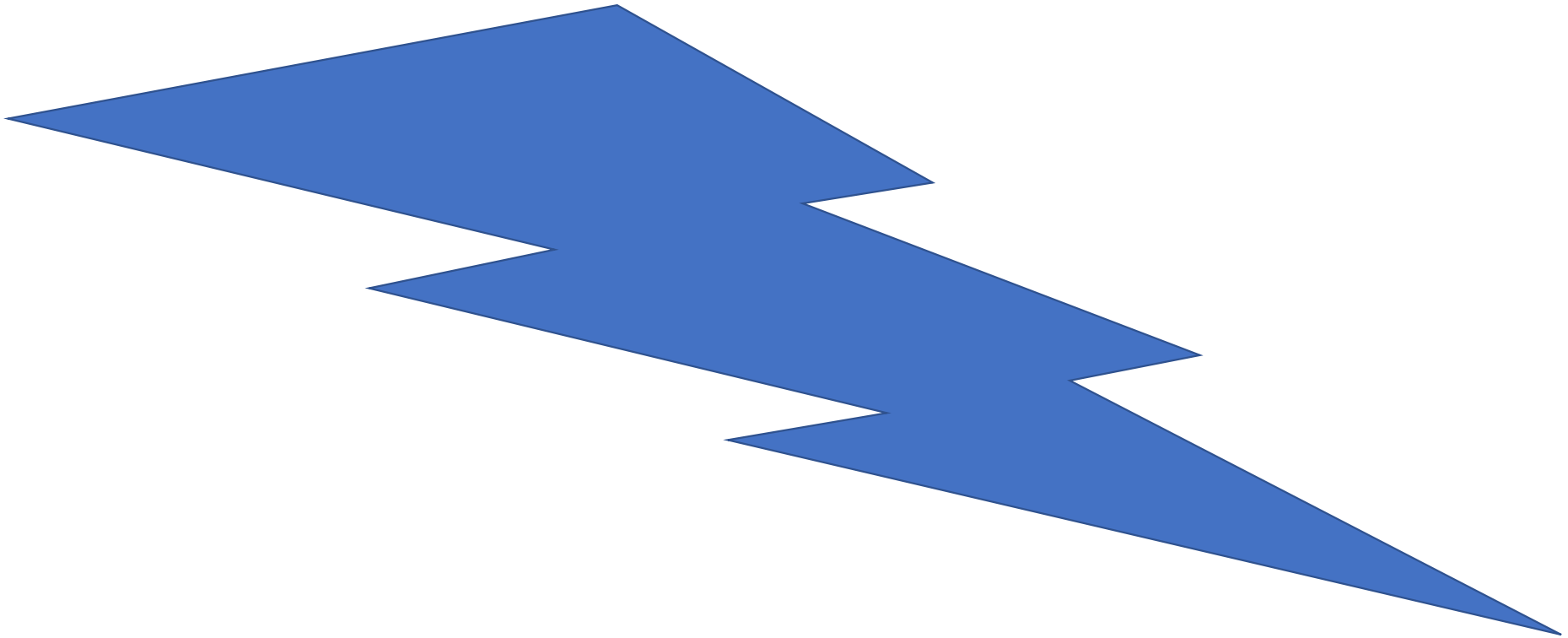
Lab03p04: String Object

```
54  const char NoOfSupportCMD = 4;
55  const char *support_cmd[] = {
56      "blink",
57      "off",
58      "on",
59      "get"
60  };
```

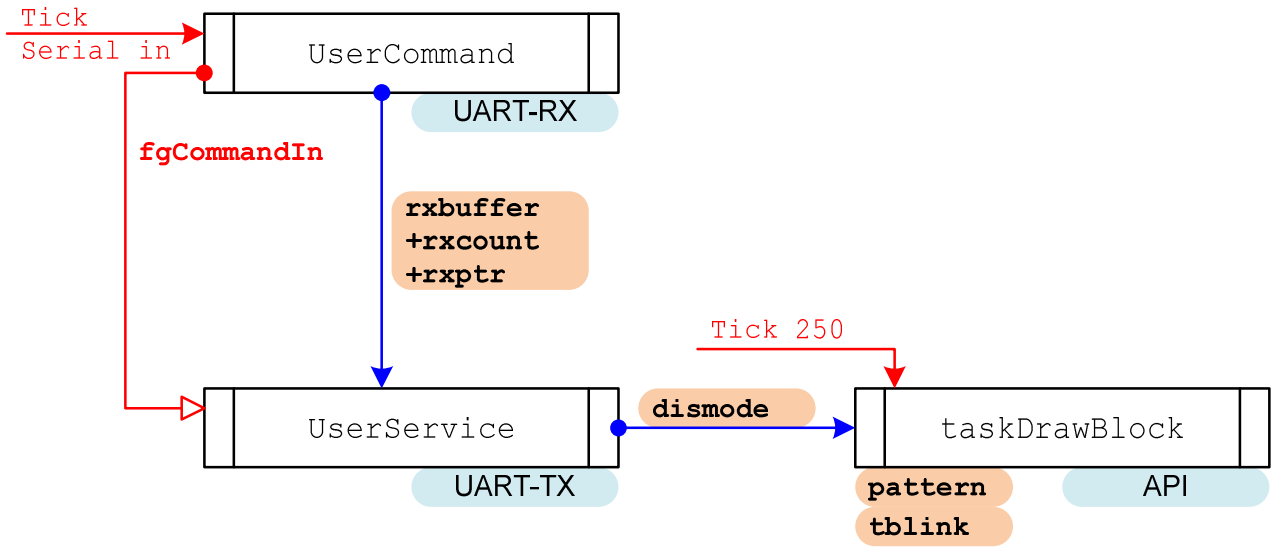
```
61  void UserService() {
62      String vcmd;
63      char i, m;
64      fgCommandIn = 0;
65
66      vcmd = String(rxbuffer);
67      m = -1;
68      i = 0;
69      while ((i < NoOfSupportCMD) && (m < 0)) {
70          if (vcmd.compareTo(support_cmd[i]) == 0)
71              m = i;
72          else
73              i++;
74      }
75      switch (m) {
76          case 0:
77          case 1:
78          case 2:
79          case 3:
80              Serial.print(": Command no. = ");
81              Serial.println(m, DEC);
82              break;
83          default:
84              Serial.println(": Invalid command");
85              break;
86      }
87      Serial.print("\n\r>");
88  }
```



L03p05

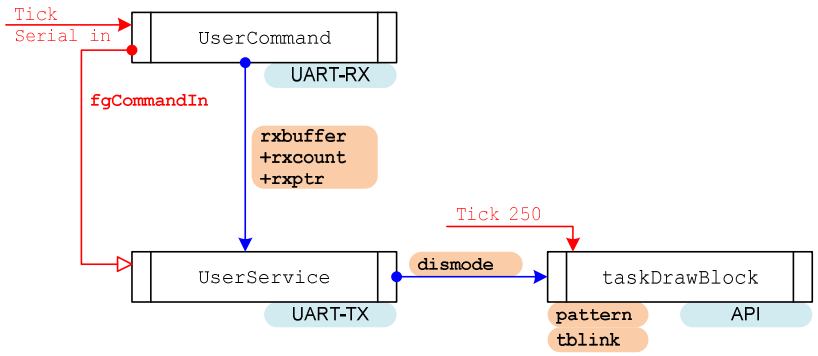


Lab03p05: Combine UCI to scheduled tasks



Lab03p05: Combine UCI to scheduled tasks

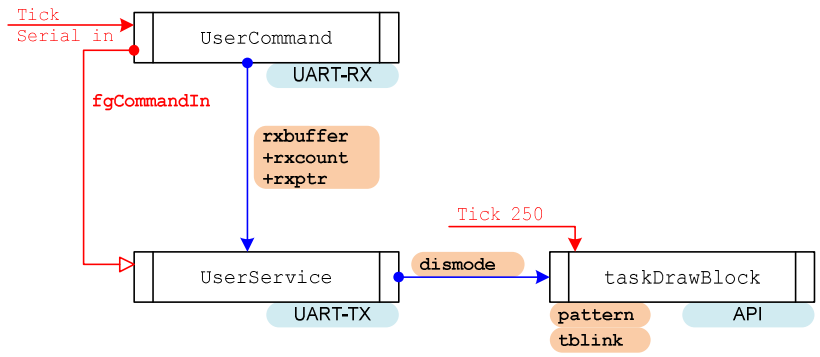
```
46  void loop() {
47      UserCommand();
48      myschedule();
49      if (rqtask == 1) {
50          rqtask = 0;
51          taskDrawBlock();
52      }
53      if (fgCommandIn == 1) UserService();
54  }
```




```
23  void setup() {
24      delay(3000);
25      // H/W initialize
26      Serial.begin(115200);
27      SPI.begin();
28      SPI.setBitOrder(MSBFIRST);
29      SPI.setClockDivider(SPI_CLOCK_DIV16);
30      SPI.setDataMode(SPI_MODE0);
31      pinMode(CS_PIN, OUTPUT);
32      digitalWrite(CS_PIN, HIGH);
33      MAX7219_init();
34      // Variables and state initialize
35      rxcount = 0;
36      rxptr = &rxbuffer[0];
37      tick_last = millis();   //for sceduler
38      rqtask = 0;
39      tblink = 0;
40      dismode = 0;
41      // End of setup
42      MAX7219_write_reg(REG_SHUTDOWN, 0x01);
43      Serial.println("\n\r\n\r\n\r#### End of setup ####");
44      Serial.print('>');
45  }
```

Lab03p05: Combine UCI to scheduled tasks

```ino
void UserService() {
    String vcmd;
    char i, m;
    fgCommandIn = 0;
    vcmd = String(rxbuffer);
    m = -1;
    i = 0;
    while ((i < NoOfSupportCMD) && (m < 0)) {
        if (vcmd.compareTo(support_cmd[i]) == 0)
            m = i;
        else
            i++;
    }
```




```ino
    switch (m) {
        case 0:
            dismode = 1;
            Serial.println(": Blinking LED");
            break;
        case 1:
            dismode = 2;
            Serial.println(": Off LED");
            break;
        case 2:
            dismode = 3;
            Serial.println(": On LED");
            break;
        case 3:
            Serial.print(": Current mode is ");
            if (dismode == 1)
                Serial.println("Blinking.");
            else if (dismode == 2)
                Serial.println("Off.");
            else
                Serial.println("On.");
            break;
        default:
            Serial.println(": Invalid command");
            break;
    }
    Serial.print("\n\r>");
}
```

Lab03p05: Combine UCI to scheduled tasks

```
75  void taskDrawBlock() {
76    int i;
77      switch (dismode)
78      {
79      case 1:
80          tblink ^= 1;
81          break;
82      case 2:
83          tblink = 0;
84          break;
85      case 3:
86          tblink = 1;
87          break;
88      }
89      if (tblink == 1) {
90          for (i = 0; i < 8; i++) {
91              MAX7219_write_reg(REG_DIGIT(i), pattern[0][i]);
92          }
93      }
94      else {
95          for (i = 0; i < 8; i++) {
96              MAX7219_write_reg(REG_DIGIT(i), 0);
97          }
98      }
99  }
```