 Instituto Politécnico Nacional

Escuela Superior de Computo

Materia:

Introducción a los microcontroladores.

Profesor:

Sanchez Aguilar Fernando

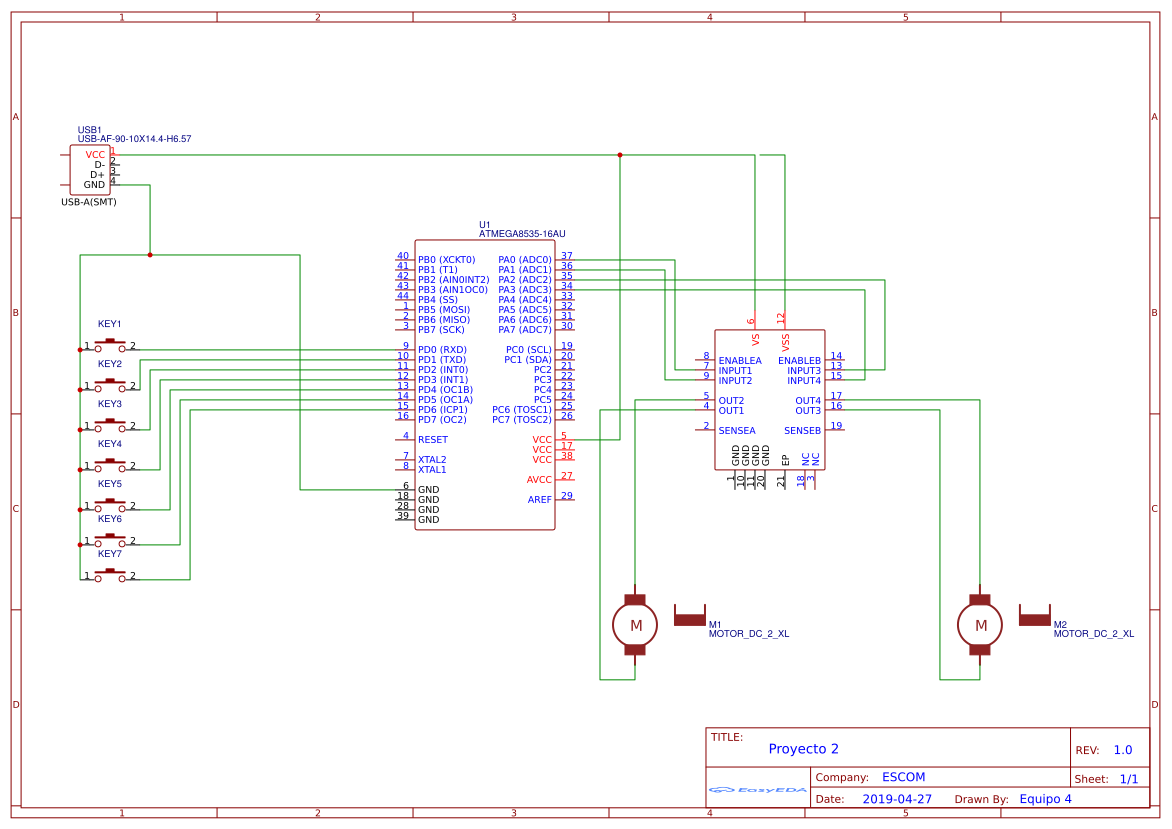
Alumnos:

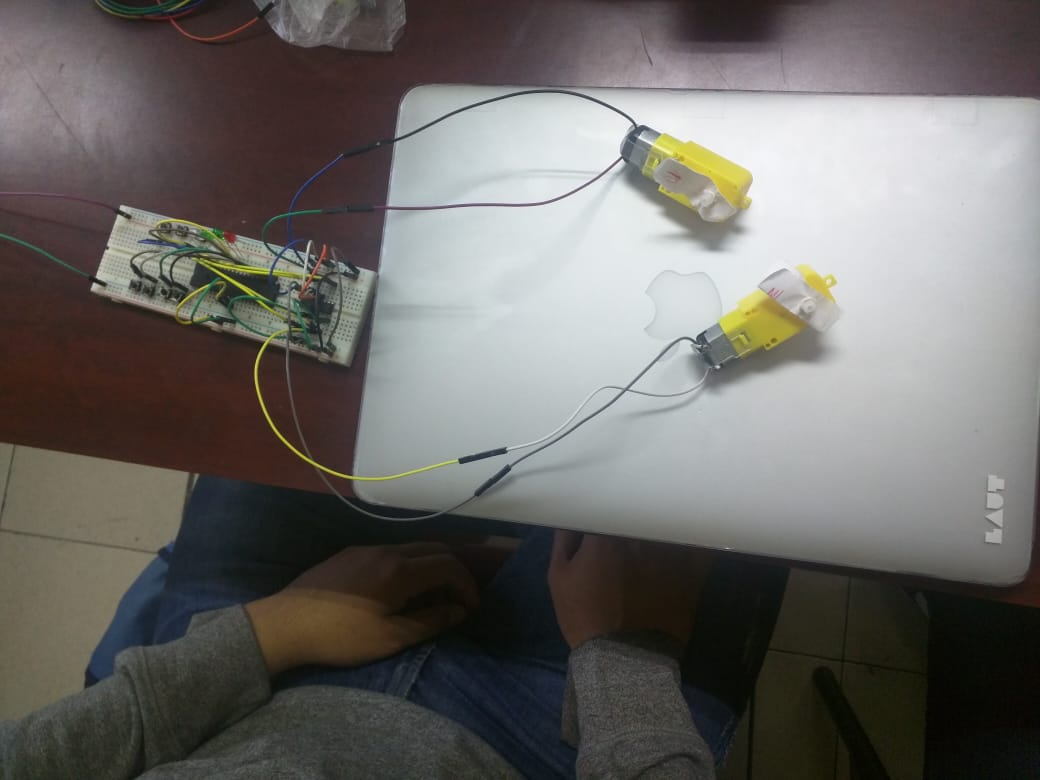
Aldavera Gallaga Iván

Lara Soto Rubén Jair

Morales Castellanos Adolfo Erik

Proyecto 2





1. /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
2. This program was created by the
3. CodeWizardAVR V2.60 Evaluation
4. Automatic Program Generator
5. © Copyright 1998-2012 Pavel Haiduc, HP InfoTech s.r.l.
6. http://www.hpinfotech.com
8. Project :
9. Version :
10. Date    : 6/3/2019
11. Author  : Equipo 4
12. Company :
13. Comments:

16. Chip type               : ATmega8515
17. Program type            : Application
18. AVR Core Clock frequency: 1.000000 MHz
19. Memory model            : Small
20. External RAM size       : 0
21. Data Stack size         : 128
22. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/
24. #include <mega8515.h>
26. #include <io.h>
27. #include <delay.h>
28. #define boton6 PINC.6
29. #define boton5 PINC.5
30. #define boton4 PINC.4
31. #define boton3 PINC.3
32. #define boton2 PINC.2
33. #define boton1 PINC.1
34. #define boton PINC.0
36. bit botonp;
37. bit botonp1;
38. bit botonp2;
39. bit botonp3;
40. bit botonp4;
41. bit botonp5;
42. bit botonp6;
44. bit botona;
45. bit botonb;
46. bit botonc;
47. bit botond;
48. bit botone;
49. bit botonf;
50. bit botong;
52. **int** control=0, control2=0, control3=0, i=0, control4=0;
53. **int** movimientos[10]={0,0,0,0,0,0,0,0,0,0};
55. **void** mover( **int** movimientos[] ){
56. **for**(i=0; i<10; i++){
57. **if** (movimientos[i]==0){
59. delay\_ms(500);
61. }
62. **else** **if**(movimientos[i]==1){  //Adelante
63. PORTA=0x0A;
64. delay\_ms(500);
65. }**else** **if**( movimientos[i]==2){   //Atras
66. PORTA=0x0A;
68. delay\_ms(500);
69. }**else** **if**( movimientos[i]==3){   //A la derecha
70. PORTA=0x08;
72. delay\_ms(500);
73. }**else** **if**( movimientos[i]==4){   //A la izquierda
74. PORTA=0x02;
76. delay\_ms(500);
77. }
78. //BOTON PAUSA
79. **if** (boton6==0)
80. botong=0;
81. **else**
82. botong=1;
83. **if** ((botonp6==1)&&(botong==0)) //hubo cambio de flanco de 1 a 0
84. {
85. PORTA=0x00;
87. **while** (control4<1)
88. {
89. //BOTON GO
90. **if** (boton4==0)
91. botone=0;
92. **else**
93. botone=1;
94. **if** ((botonp4==1)&&(botone==0)) //hubo cambio de flanco de 1 a 0
95. {
96. control4=1;
97. }
98. **if** ((botonp4==0)&&(botone==1)) //hubo cambio de flanco de 0 a 1
99. delay\_ms(30);
100. botonp4=botone;
101. }
102. control4=0;
103. }
104. **if** ((botonp6==0)&&(botong==1)) //hubo cambio de flanco de 0 a 1
105. delay\_ms(5);
106. botonp6=botong;
107. }
108. PORTA=0x00;
109. }
111. **void** main(**void**)
112. {
113. // Declare your local variables here
115. // Input/Output Ports initialization
116. // Port A initialization
117. // Function: Bit7=In Bit6=In Bit5=In Bit4=In Bit3=Out Bit2=Out Bit1=Out Bit0=Out
118. DDRA=(0<<DDA7) | (0<<DDA6) | (0<<DDA5) | (0<<DDA4) | (1<<DDA3) | (1<<DDA2) | (1<<DDA1) | (1<<DDA0);
119. // State: Bit7=T Bit6=T Bit5=T Bit4=T Bit3=0 Bit2=0 Bit1=0 Bit0=0
120. PORTA=(0<<PORTA7) | (0<<PORTA6) | (0<<PORTA5) | (0<<PORTA4) | (0<<PORTA3) | (0<<PORTA2) | (0<<PORTA1) | (0<<PORTA0);
122. // Port B initialization
123. // Function: Bit7=In Bit6=In Bit5=In Bit4=In Bit3=In Bit2=In Bit1=In Bit0=In
124. DDRB=(0<<DDB7) | (0<<DDB6) | (0<<DDB5) | (0<<DDB4) | (0<<DDB3) | (0<<DDB2) | (0<<DDB1) | (0<<DDB0);
125. // State: Bit7=T Bit6=T Bit5=T Bit4=T Bit3=T Bit2=T Bit1=T Bit0=T
126. PORTB=(0<<PORTB7) | (0<<PORTB6) | (0<<PORTB5) | (0<<PORTB4) | (0<<PORTB3) | (0<<PORTB2) | (0<<PORTB1) | (0<<PORTB0);
128. // Port C initialization
129. // Function: Bit7=In Bit6=In Bit5=In Bit4=In Bit3=In Bit2=In Bit1=In Bit0=In
130. DDRC=(0<<DDC7) | (0<<DDC6) | (0<<DDC5) | (0<<DDC4) | (0<<DDC3) | (0<<DDC2) | (0<<DDC1) | (0<<DDC0);
131. // State: Bit7=P Bit6=P Bit5=P Bit4=P Bit3=P Bit2=P Bit1=P Bit0=P
132. PORTC=(1<<PORTC7) | (1<<PORTC6) | (1<<PORTC5) | (1<<PORTC4) | (1<<PORTC3) | (1<<PORTC2) | (1<<PORTC1) | (1<<PORTC0);
134. // Port D initialization
135. // Function: Bit7=In Bit6=In Bit5=In Bit4=In Bit3=In Bit2=In Bit1=In Bit0=In
136. DDRD=(0<<DDD7) | (0<<DDD6) | (0<<DDD5) | (0<<DDD4) | (0<<DDD3) | (0<<DDD2) | (0<<DDD1) | (0<<DDD0);
137. // State: Bit7=T Bit6=T Bit5=T Bit4=T Bit3=T Bit2=T Bit1=T Bit0=T
138. PORTD=(0<<PORTD7) | (0<<PORTD6) | (0<<PORTD5) | (0<<PORTD4) | (0<<PORTD3) | (0<<PORTD2) | (0<<PORTD1) | (0<<PORTD0);
140. // Port E initialization
141. // Function: Bit2=In Bit1=In Bit0=In
142. DDRE=(0<<DDE2) | (0<<DDE1) | (0<<DDE0);
143. // State: Bit2=T Bit1=T Bit0=T
144. PORTE=(0<<PORTE2) | (0<<PORTE1) | (0<<PORTE0);
146. // Timer/Counter 0 initialization
147. // Clock source: System Clock
148. // Clock value: Timer 0 Stopped
149. // Mode: Normal top=0xFF
150. // OC0 output: Disconnected
151. TCCR0=(0<<PWM0) | (0<<COM01) | (0<<COM00) | (0<<CTC0) | (0<<CS02) | (0<<CS01) | (0<<CS00);
152. TCNT0=0x00;
153. OCR0=0x00;
155. // Timer/Counter 1 initialization
156. // Clock source: System Clock
157. // Clock value: Timer1 Stopped
158. // Mode: Normal top=0xFFFF
159. // OC1A output: Disconnected
160. // OC1B output: Disconnected
161. // Noise Canceler: Off
162. // Input Capture on Falling Edge
163. // Timer1 Overflow Interrupt: Off
164. // Input Capture Interrupt: Off
165. // Compare A Match Interrupt: Off
166. // Compare B Match Interrupt: Off
167. TCCR1A=(0<<COM1A1) | (0<<COM1A0) | (0<<COM1B1) | (0<<COM1B0) | (0<<WGM11) | (0<<WGM10);
168. TCCR1B=(0<<ICNC1) | (0<<ICES1) | (0<<WGM13) | (0<<WGM12) | (0<<CS12) | (0<<CS11) | (0<<CS10);
169. TCNT1H=0x00;
170. TCNT1L=0x00;
171. ICR1H=0x00;
172. ICR1L=0x00;
173. OCR1AH=0x00;
174. OCR1AL=0x00;
175. OCR1BH=0x00;
176. OCR1BL=0x00;
178. // Timer(s)/Counter(s) Interrupt(s) initialization
179. TIMSK=(0<<TOIE1) | (0<<OCIE1A) | (0<<OCIE1B) | (0<<TICIE1) | (0<<TOIE0) | (0<<OCIE0);
181. // External Interrupt(s) initialization
182. // INT0: Off
183. // INT1: Off
184. // INT2: Off
185. MCUCR=(0<<SRE) | (0<<SRW10) | (0<<ISC11) | (0<<ISC10) | (0<<ISC01) | (0<<ISC00);
186. EMCUCR=(0<<SRL2) | (0<<SRL1) | (0<<SRL0) | (0<<SRW01) | (0<<SRW00) | (0<<SRW11) | (0<<ISC2);
188. // USART initialization
189. // USART disabled
190. UCSRB=(0<<RXCIE) | (0<<TXCIE) | (0<<UDRIE) | (0<<RXEN) | (0<<TXEN) | (0<<UCSZ2) | (0<<RXB8) | (0<<TXB8);
192. // Analog Comparator initialization
193. // Analog Comparator: Off
194. ACSR=(1<<ACD) | (0<<ACBG) | (0<<ACO) | (0<<ACI) | (0<<ACIE) | (0<<ACIC) | (0<<ACIS1) | (0<<ACIS0);
196. // SPI initialization
197. // SPI disabled
198. SPCR=(0<<SPIE) | (0<<SPE) | (0<<DORD) | (0<<MSTR) | (0<<CPOL) | (0<<CPHA) | (0<<SPR1) | (0<<SPR0);

201. **while** (1)
202. {
204. //BOTON GO
205. **if** (boton4==0)
206. botone=0;
207. **else**
208. botone=1;
209. **if** ((botonp4==1)&&(botone==0)) //hubo cambio de flanco de 1 a 0
210. {
211. control=3;
212. }
213. **if** ((botonp4==0)&&(botone==1)) //hubo cambio de flanco de 0 a 1
214. delay\_ms(30);
215. botonp4=botone;
217. //BOTON CLEAR
218. **if** (boton5==0)
219. botonf=0;
220. **else**
221. botonf=1;
222. **if** ((botonp5==1)&&(botonf==0)) //hubo cambio de flanco de 1 a 0
223. {
224. control=1;
225. }
226. **if** ((botonp5==0)&&(botonf==1)) //hubo cambio de flanco de 0 a 1
227. delay\_ms(30);
228. botonp5=botonf;
230. **if**(control==0)
231. {
232. **while**(control2<1){
233. //Avanza adelante
234. **if** (boton==0)
235. botona=0;
236. **else**
237. botona=1;
238. **if** ((botonp==1)&&(botona==0)) //hubo cambio de flanco de 1 a 0
239. {
241. movimientos[control]=1;
242. control++;
243. }
244. **if** ((botonp==0)&&(botona==1)) //hubo cambio de flanco de 0 a 1
245. delay\_ms(30);
246. botonp=botona;
247. //Boton atras
248. **if** (boton1==0)
249. botonb=0;
250. **else**
251. botonb=1;
252. **if** ((botonp1==1)&&(botonb==0)) //hubo cambio de flanco de 1 a 0
253. {
255. movimientos[control]=2;
256. control++;
257. }
258. **if** ((botonp1==0)&&(botonb==1)) //hubo cambio de flanco de 0 a 1
259. delay\_ms(30);
260. botonp1=botonb;
262. //Avanza derecha
263. **if** (boton2==0)
264. botonc=0;
265. **else**
266. botonc=1;
267. **if** ((botonp2==1)&&(botonc==0)) //hubo cambio de flanco de 1 a 0
268. {
270. movimientos[control]=3;
271. control++;
272. }
273. **if** ((botonp2==0)&&(botonc==1)) //hubo cambio de flanco de 0 a 1
274. delay\_ms(30);
275. botonp2=botonc;
277. //Avanza izquierda
278. **if** (boton3==0)
279. botond=0;
280. **else**
281. botond=1;
282. **if** ((botonp3==1)&&(botond==0)) //hubo cambio de flanco de 1 a 0
283. {
285. movimientos[control]=4;
286. control++;
287. }
288. **if** ((botonp3==0)&&(botond==1)) //hubo cambio de flanco de 0 a 1
289. delay\_ms(30);
290. botonp3=botond;
292. **if**(control==10){
293. control2=1;
294. control=10;
295. }
296. }
297. control2=0;
298. }**else** **if**(control==1){
300. **while**(control3<8){
302. movimientos[control]=0;
303. control3++;
304. }
305. control3=0;
306. control=0;
307. }**else** **if**(control==3){
308. mover(movimientos);
309. control=10;
310. }

313. }
314. }