```
;       Filename:               RCXEEP.lib
;       Version:                1.10
;       Date:                   20. marts 2007
;       Author:                 Ib Refer
;
;       Beskrivelse:
;               Dette bibliotek indeholder funktioner der kan genkende kommandoer afsendt i følgende formater.
;               *       Sdandard RC5 protokol fra Philips
;               *       Extended RC5 protokol fra Philips
;               *       Sdandart RC6 protokol fra Philips
;               Endvidere indeholder biblioteket nogle funktioner til at læse og skrive i PIC'ens EEprom (data-område)
;
;       Funktioner (IR):
;
;               wait4RC         Denne funktion venter på en kode fra en af de 3 protokoller,
;                               når den returnerer findes informationen om koden i følgende registre:
;                               IRtype      Dette register kan antage værdien 5 eller 6, som angiver formatet
;                               IRadr       Dette register angiver adressen også kaldet device-nr.
;                               IRdata      Dette register indeholder selve komandoen
;                               IRmode      Dette register angiver mode for RC6, og er altid 0 for RC5
;                               "stdRC5"    Dette flag er "1" for standard RC5 og "0" for extended RC5
;                               "IRerr"     Dette flag er "0" for rigtigt modtaget kode og  "1" ved fejl
;                               "newkey"    Dette flag er "1" når en ny kanp trykkes, holdes en knap nede bliver flaget "0"
;
;               wait4noRC       Denne funktion skipper igangværende signaler og venter til,
;                               der ikke har været noget signal aktivitet i ca. 2,66ms
;
;       Funktioner (EEprom):
;
;               wrEEp           Denne funktion skriver en byte til EEprom området.
;                               NB! Adressen skal ligge i adrEE-reg. og data i dataEE-reg. inden funktionen kaldes
;
;               rdEEp           Denne funktion læser en byte fra EEprom området.
;                               NB! Adressen skal ligge i adrEE-reg. inden funktionen kaldes
;                               Resultatet af læsningen ligger i W-reg. når funktionen returnerer
;
;               compareEE       Denne funktion sammenligner aktuelle RC-oplysninger med nogle, der er gemt i EEprom'en
;                               disse informationer fylder 4 bytes (type, mode, adr, data). Man skal angive en start adresse, på
;                               den første af de 4 bytes i EEprom'en som man vil sammenligne med aktuel RC-info. Denne
;                               adresse værdi skal ligge i W-reg inden funktionen kaldes.
;                               Z = "1" hvis der er et match,  ellers er Z = "0" når der returneres fra funktionen.
;
;               save2EE         Denne funktion gemmer aktuelle RC-oplysninger i EEprom'en (fylder 4 bytes)
;                               Man skal angive en start adresse, som svarer til den første af de 4 bytes i EEprom'en som
;                               man vil gemme aktuel RC-info i. Denne adresse værdi skal ligge i W-reg inden funktionen
;                               kaldes.
;-----------------------------------------------------------------------------------------------------------------RC-Routiner
;Denne routine anvender bit-historikken til at springe til den rigtige funktion i RC5
RDbit_5         ANDLW   B'00000011'     ;mask de to seneste bit (historikken)
                ADDWF   PCL,F           ;gå til den routine, der passer til masken
                GOTO    rd00_5          ;udfør denne hvis der var et 0 og efterfulgt at endnu et 0
                GOTO    rd01_5          ;udfør denne hvis der var et 0 og efterfulgt at et 1
                GOTO    rd10_5          ;udfør denne hvis der var et 1 og efterfulgt at et 0
                GOTO    rd11_5          ;udfør denne hvis der var et 1 og efterfulgt at endnu et 1
; Denne routine anvender bit-historikken til at springe til den rigtige funktion i RC6
RDbit_6         ANDLW   B'00000011'     ;mask de to seneste bit (historikken)
                ADDWF   PCL,F           ;gå til den routine, der passer til masken
                GOTO    rd00_6          ;udfør denne hvis der var et 0 og efterfulgt at endnu et 0
                GOTO    rd01_6          ;udfør denne hvis der var et 0 og efterfulgt at et 1
                GOTO    rd10_6          ;udfør denne hvis der var et 1 og efterfulgt at et 0
                GOTO    rd11_6          ;udfør denne hvis der var et 1 og efterfulgt at endnu et 1
;"SKIP HIGH RC6" Denne routine skal bruges hvis vi havde et "0" og fik endnu et "0", altså endnu en kort puls (lav)
rd00_6          CLRF    cnt             ;reset counter
Hloop00_6       CALL    inccnt_6        ;inc, counter
                BTFSC   STATUS,Z        ;is there overflow?
                GOTO    IRerror         ;if yes: go to error handler
                BTFSC   IRinput         ;is puls high?
                GOTO    Hloop00_6       ;if yes: stay in high-loop
;"READ LOW RC6" Denne routine skal bruges hvis vi havde et "0" og  fik et "1", altså en lang puls (lav)
rd10_6          CLRF    cnt             ;reset counter
Lloop10_6       CALL    inccnt_6        ;inc, counter
                BTFSC   STATUS,Z        ;is there overflow?
```

```
 73:                 GOTO    IRerror         ;if yes: go to error handler
 74:                 BTFSS   IRinput         ;is puls low?
 75:                 GOTO    Lloop10_6       ;if yes: stay in low-loop
 76:                 MOVLW   Lpuls_6         ;\ cnt - puls:
 77:                 SUBWF   cnt,W           ;/ carry = 1 if long puls / carry = 0 if short puls
 78:                 RETURN                  ;
 79: ;"SKIP LOW RC6" Denne routine skal bruges hvis vi havde et "1" og fik endnu et "1", altså endnu en kort puls (høj)
 80: rd11_6          CLRF    cnt             ;reset counter
 81: Lloop11_6       CALL    inccnt_6        ;inc, counter
 82:                 BTFSC   STATUS,Z        ;is there overflow?
 83:                 GOTO    IRerror         ;if yes: go to error handler
 84:                 BTFSS   IRinput         ;is puls low?
 85:                 GOTO    Lloop11_6       ;if yes: stay in low-loop
 86: ;"READ HIGH RC6" Denne routine skal bruges hvis vi havde et "1" og  fik et "0", altså en lang puls (høj)
 87: rd01_6          CLRF    cnt             ;reset counter
 88: Hloop01_6       CALL    inccnt_6        ;inc, counter
 89:                 BTFSC   STATUS,Z        ;is there overflow?
 90:                 GOTO    IRerror         ;if yes: go to error handler
 91:                 BTFSC   IRinput         ;is puls high?
 92:                 GOTO    Hloop01_6       ;if yes: stay in high-loop
 93:                 MOVF    cnt,W           ;\ puls - cnt:
 94:                 SUBLW   Hpuls_6         ;/ carry = 0 if long puls / carry = 1 if short puls
 95:                 RETURN
 96: ;"SKIP LOW RC5" Denne routine skal bruges hvis vi havde et "0" og fik endnu et "0", altså endnu en kort puls (høj)
 97: rd00_5          CLRF    cnt             ;reset counter
 98: Lloop00_5       CALL    inccnt_5        ;inc, counter
 99:                 BTFSC   STATUS,Z        ;is there overflow?
100:                 GOTO    IRerror         ;if yes: go to error handler
101:                 BTFSS   IRinput         ;is puls low?
102:                 GOTO    Lloop00_5       ;if yes: stay in high-loop
103: ;"READ HIGH RC5" Denne routine skal bruges hvis vi havde et "0" og  fik et "1", altså en lang puls (høj)
104: rd10_5          CLRF    cnt             ;reset counter
105: Hloop10_5       CALL    inccnt_5        ;inc, counter
106:                 BTFSC   STATUS,Z        ;is there overflow?
107:                 GOTO    IRerror         ;if yes: go to error handler
108:                 BTFSC   IRinput         ;is puls high?
109:                 GOTO    Hloop10_5       ;if yes: stay in low-loop
110:                 MOVLW   Hpuls_5         ;\ puls-cnt:
111:                 SUBWF   cnt,W           ;/ carry = 0 if long puls / carry = 1 if short puls
112:                 RETURN                  ;
113: ;"SKIP HIGH RC5" Denne routine skal bruges hvis vi havde et "1" og fik endnu et "1", altså endnu en kort puls (lav)
114: rd11_5          CLRF    cnt             ;reset counter
115: Hloop11_5       CALL    inccnt_5        ;inc, counter
116:                 BTFSC   STATUS,Z        ;is there overflow?
117:                 GOTO    IRerror         ;if yes: go to error handler
118:                 BTFSC   IRinput         ;is puls high?
119:                 GOTO    Hloop11_5       ;if yes: stay in low-loop
120: ;"READ LOW RC5" Denne routine skal bruges hvis vi havde et "1" og  fik et "0", altså en lang puls (lav)
121: rd01_5          CLRF    cnt             ;reset counter
122: Lloop01_5       CALL    inccnt_5        ;inc, counter
123:                 BTFSC   STATUS,Z        ;is there overflow?
124:                 GOTO    IRerror         ;if yes: go to error handler
125:                 BTFSS   IRinput         ;is puls low?
126:                 GOTO    Lloop01_5       ;if yes: stay in high-loop
127:                 MOVF    cnt,W           ;\ puls - cnt:
128:                 SUBLW   Lpuls_5         ;/ carry = 1 if long puls / carry = 0 if short puls
129:                 RETURN
130: ; Denne routine tæller counteren "cnt" op, counter-værdien anvendes til at skælne korte og lange pulser i RC6
131: inccnt_6        NOP                     ;denne routine tager 6,4us i de loop's hvor den bruges,
132:                 NOP                     ;når der anvendes et 10MHz krystal
133:                 NOP
134:                 NOP
135:                 NOP
136:                 NOP
137:                 INCF    cnt,F           ;inc, counter
138:                 RETURN
139: ; Denne routine tæller counteren "cnt" op, counter-værdien anvendes til at skælne korte og lange pulser i RC5
140: inccnt_5        NOP                     ;denne routine tager 10us i de loop's hvor den bruges,
141:                 NOP                     ;når der anvendes et 10MHz krystal
142:                 NOP
143:                 NOP
144:                 NOP
```

```
145:                    NOP
146:                    NOP
147:                    NOP
148:                    NOP
149:                    NOP
150:                    NOP
151:                    NOP
152:                    NOP
153:                    NOP
154:                    NOP
155:                    INCF        cnt,F               ;inc, counter
156:                    RETURN                          ;
157: ; Denne routine læser et tryk på fjernbetjeningen vfr. RC6 protokollen
158: RC6read            MOVLW       d'3'                ;\ gør klar til at læse
159:                    MOVWF       bitcnt              ;/ 3 IRmode-bit's
160: RDmode_6           movf        IRmode,W            ;hent historikken for bit-strømmen
161:                    CALL        RDbit_6             ;læs IRmode-bit
162:                    RLF         IRmode,F            ;skift det ind i IRmode-reg.
163:                    BTFSC       IRerr               ;\ Hvis der var fejl, så skip
164:                    GOTO        unknownRC           ;/ resten af datastrømmen...
165:                    DECFSZ      bitcnt,F            ;er der læst 3 bit's?
166:                    GOTO        RDmode_6            ;hvis nej: så bliv i loopet
167: RDtoggle_6         movf        IRmode,W            ;hent historikken for bit-strømmen
168:                    CALL        RDbit_6             ;læs toggle-bit
169:                    RLF         toggle,F            ;skift det ind i toggle-reg.
170:                    BTFSC       IRerr               ;\ Hvis der var fejl, så skip
171:                    GOTO        unknownRC           ;/ resten af datastrømmen...
172:                    MOVLW       b'111'              ;\  mask mode-reg. så det kun
173:                    ANDWF       IRmode,F            ;/ indeholder de bit vi har læst
174:                    BTFSC       toggle,0            ;\ hvis toggle-bit'et var 1
175:                    MOVLW       b'01'               ;/ så sæt historikken til "01"
176:                    BTFSS       toggle,0            ;\ hvis toggle-bit'et var 0
177:                    MOVLW       b'10'               ;/ så sæt historikken til "10"
178:                    MOVWF       IRadr               ;gem historikken i IRadr-reg.
179:                    BTFSC       toggle,0            ;\ hvis toggle-bit'et var 1
180:                    MOVLW       D'75'               ;/ så sæt en kort delay-værdi
181:                    BTFSS       toggle,0            ;\ hvis toggle-bit'et var 0
182:                    MOVLW       D'223'              ;/ så sæt en lang delay-værdi
183:                    MOVWF       cnt                 ;gem delay-værdi
184:                    CALL        Tdelay              ;udfør det ønskede delay
185:                    MOVLW       d'8'                ;\ gør klar til at læse
186:                    MOVWF       bitcnt              ;/ 8 IRadr-bit's
187: RDadr_6            movf        IRadr,W             ;hent historikken for bit-strømmen
188:                    CALL        RDbit_6             ;læs IRadr-bit
189:                    RLF         IRadr,f             ;skift det ind i IRadr-reg.
190:                    BTFSC       IRerr               ;\ Hvis der var fejl, så skip
191:                    GOTO        unknownRC           ;/ resten af datastrømmen...
192:                    DECFSZ      bitcnt,F            ;er der læst 8 bit's?
193:                    GOTO        RDadr_6             ;hvis nej: så bliv i loopet
194:                    MOVF        IRadr,W             ;\ flyt historikken for bit-strømmen
195:                    MOVWF       IRdata              ;/ over i IRdata
196:                    MOVLW       d'8'                ;\ gør klar til at læse
197:                    MOVWF       bitcnt              ;/ 8 IRdata-bit's
198: RDdata_6           movf        IRdata,W            ;hent historikken for bit-strømmen
199:                    CALL        RDbit_6             ;læs fIRdata-bit
200:                    RLF         IRdata,f            ;skift det ind i IRdata-reg.
201:                    BTFSC       IRerr               ;\ Hvis der var fejl, så skip
202:                    GOTO        unknownRC           ;/ resten af datastrømmen...
203:                    DECFSZ      bitcnt,F            ;er der læst 8 bit's?
204:                    GOTO        RDdata_6            ;hvis nej: så bliv i loopet
205:                    MOVLW       D'6'                ;\ sæt IRtypen til
206:                    MOVWF       IRtype              ;/ RC6
207:                    CALL        keyupdate           ;optater "newkey" flaget
208:                    RETURN
209: ; Denne routine læser et tryk på fjernbetjeningen vfr. RC5 protokollen
210: RC5read            BSF         stdRC5              ;sæt standard RC5 protokol
211:                    MOVLW       b'11'               ;\ sæt bit-historikken (11)
212:                    MOVWF       IRadr               ;/ så der læses rigtigt
213:                    GOTO        RC5start            ; begund læsning af et tastetryk
214: RC5read1           CLRF        IRadr               ; sæt bit-historikken (00), så det læses rigtigt - special RC5 protokol
215:                    BCF         stdRC5              ;sæt extended RC5 protokol
216: RC5start           MOVLW       d'6'                ;\ gør klar til at læse 6 bit's
```

```
217:                MOVWF   bitcnt          ;/ 1 toggle-bit og 5 IRadr-bit's
218: RDadr_5        MOVF    IRadr,W         ;hent historikken for bit-strømmen
219:                CALL    RDbit_5         ;læs IRadr-bit
220:                RLF     IRadr,f         ;skift det ind i IRadr-reg.
221:                BTFSC   IRerr           ;\ Hvis der var fejl, så skip
222:                GOTO    unknownRC       ;/ resten af datastrømmen...
223:                DECFSZ  bitcnt,F        ;er der læst 6 bit's?
224:                GOTO    RDadr_5         ;hvis nej: så bliv i loopet
225:                MOVF    IRadr,W         ;\
226:                MOVWF   IRdata          ;  flyt historikken for bit-strømmen
227:                MOVLW   b'11'           ;  (de 2 LSB) over i IRdata
228:                ANDWF   IRdata          ;/
229:                MOVLW   d'6'            ;\ gør klar til at læse
230:                MOVWF   bitcnt          ;/ 6 IRdata-bit's
231: RDdata_5       movf    IRdata,W        ;hent historikken for bit-strømmen
232:                CALL    RDbit_5         ;læs fIRdata-bit
233:                RLF     IRdata,f        ;skift det ind i IRdata-reg.
234:                BTFSC   IRerr           ;\ Hvis der var fejl, så skip
235:                GOTO    unknownRC       ;/ resten af datastrømmen...
236:                DECFSZ  bitcnt,F        ;er der læst 6 bit's?
237:                GOTO    RDdata_5        ;hvis nej: så bliv i loopet
238:                MOVLW   D'5'            ;\ sæt IRtype til
239:                MOVWF   IRtype          ;/ RC5...
240:                btfss   IRadr,5         ;\
241:                MOVLW   d'0'            ;
242:                BTFSC   IRadr,5         ;  hent toggle-bit'et ud af IRadr-reg. og gem det i Toggle-reg.
243:                MOVLW   d'1'            ;
244:                MOVWF   toggle          ;/
245:                MOVLW   b'00011111'     ;\ mask kun adresse informationer
246:                ANDWF   IRadr,F         ;/ og gem dem i IRadr-reg.
247:                BTFSS   stdRC5          ;\ hvis det er en extended RC5
248:                BSF     IRdata,6        ;/ så sæt bit nr 6 i IRdata-reg.
249:                CLRF    IRmode          ;nulstil IRmode
250:                CALL    keyupdate       ;optater "newkey" flaget
251:                MOVF    IRdata,w        ;\
252:                SUBLW   0xFF            ; hvis data er 255, så er det en fejl...!
253:                btfss   STATUS,Z        ;/
254:                RETURN
255: ; Denne routine anvendes hvis en puls bliver for lang
256: IRerror        bsf     IRerr           ;sæt error flaget
257:                RETURN
258: ; Denne routine sætter "newkey" flaget hvis toggle bit'et har ændret sig, "newkey" er clearet hvis toggle bit'et er det samme
259: keyupdate      MOVF    flags,w         ;\ hent oldtoggle flaget
260:                ANDLW   b'1'            ;/ ind i W-reg.
261:                XORWF   toggle,w        ;test om toggle er forskellig fra oldtoggle
262:                BTFSS   status,z        ;\
263:                BSF     newkey          ;  sæt newkey
264:                BTFSC   status,z        ;  udfra resultatet
265:                BCF     newkey          ;/
266:                BTFSS   toggle,0        ;\
267:                BCF     oldtoggle       ;  kopier toggle til
268:                BTFSC   toggle,0        ;  oldtoggle
269:                BSF     oldtoggle       ;/
270:                return
271: ; Denne routine anvendes til at udsætte en læsning af IR i forb. med toggle-bittet i RC6
272: Tcheck         CLRF    cnt             ;< når denne routine kaldes, fås et delay på ca. 1,33ms
273: Tdelay         NOP                     ;dette delay er på 5,2us x værdien af "cnt"
274:                NOP
275:                NOP
276:                NOP
277:                NOP
278:                NOP
279:                NOP
280:                NOP
281:                NOP
282:                NOP
283:                NOP
284:                DECFSZ  cnt,F
285:                GOTO    Tdelay
286:                RETURN
287: ; Denne routine anvendes i forb. med RC6 protokollen til at opsætte startbetingelser
288: init_reg       MOVLW   b'11'
```

```
289:                MOVWF   IRmode
290:                CLRF    toggle
291:                RETURN
292: ; Denne routine venter på at der kommer et tastetryk på fjernbetjeningen
293: wait4RC         BCF     IRerr           ;nulstil IRerr flaget
294: w4rc1          BTFSC   IRinput         ;\ vent på en lav
295:                GOTO    w4rc1           ;/ puls på IRinput
296:                CALL    Tcheck          ;vent 3/4 bit-bredde for RC5
297:                BTFSC   IRinput         ;hvis pilsen stadig er lav, så spring standard RC5 over
298:                GOTO    RC5read         ;hvis pulsen er høj, så gå til standard RC5
299:                CALL    RD10_6          ;Vent på at pulsen bliver høj...
300:                BTFSC   IRerr           ;\ hvis pulsen er for lang (overflow)
301:                GOTO    unknownRC       ;/ så gå til ukendt format...
302:                MOVF    cnt,w           ;\
303:                SUBLW   d'100'          ;  test om pulsen er kort !
304:                BTFSC   STATUS,C        ;/
305:                GOTO    RC5read1        ;Hvis ja: så gå til special RC5
306:                CALL    init_reg        ;Hvis nej: så opsæt startbetingelser for RC6
307: w4rc2          BTFSC   IRinput         ;\ vent på en lav
308:                GOTO    w4rc2           ;/ puls på IRinput
309:                GOTO    RC6read         ;... og start så RC6
310: unknownRC      RETURN                  ;afslut uden at gøre noget (error flaget er sat)
311: ; denne routine venter til der ikke har været noget signal aktivitet i ca. 2,66ms
312: wait4noRC      CLRF    cnt             ;clear counter
313: w4nrcL         CALL    inccnt_5        ;inc, counter (loop starts here)
314:                NOP                     ;(timing adjustment)
315:                BTFSS   IRinput         ;is puls low?
316:                GOTO    wait4noRC       ;yes: restart this procedure
317:                BTFSC   STATUS,Z        ;is there overflow?
318:                RETURN                  ;yes: then exit
319:                GOTO    w4nrcL          ;no: stay in loop
320: ;denne routine returnerer Z=1 for idle og Z=0 for busy
321: idleRC         CLRF    cnt             ;clear counter
322: ircL           CALL    inccnt_5        ;inc, counter (loop starts here)
323:                BTFSS   STATUS,Z
324:                RETURN
325:                BTFSS   IRinput
326:                RETURN
327:                GOTO    ircL
328: ;-----------------------------------------------------------------------------------------------------------EEprom-Routiner
329: ;Denne routine skriver en byte til EEprom området. Adresse skal ligge i adrEE og data i dataEE inden kaldet
330: wrEEp          BSF     STATUS,RP1      ;\  Bank 3
331:                BSF     STATUS,RP0      ;/
332:                BTFSC   EECON1,WR       ;Wait for write
333:                GOTO    $-1             ;to complete
334:                BCF     STATUS,RP0      ;bank 2
335:                MOVF    adrEE,W         ;EEmenory
336:                MOVWF   EEADR           ;adress to write...
337:                MOVF    dataEE,W        ;EEmenory
338:                MOVWF   EEDATA          ;data to write...
339:                BSF     STATUS,RP0      ;bank 3
340:                BCF     EECON1,EEPGD    ;point to EEprom menory
341:                BSF     EECON1,WREN     ;enable writes
342:                BCF     INTCON,GIE      ;disable interrupts
343:                MOVLW   0x55            ;\ Write
344:                MOVWF   EECON2          ;/ 55h
345:                MOVLW   0xAA            ;\ Write
346:                MOVWF   EECON2          ;/ AAh
347:                BSF     EECON1,WR       ;set WR bit to begin write
348:                BSF     INTCON,GIE      ;enable interrupts
349:                BCF     EECON1,WREN     ;disable writes
350:                BCF     STATUS,RP1      ;\  Bank 0
351:                BCF     STATUS,RP0      ;/
352:                RETURN
353: ;Denne routine læser en byte fra EEprom området. Adresse skal ligge i adrEE inden kaldet
354: rdEEp          BSF     STATUS,RP1      ;\  Bank 2
355:                BCF     STATUS,RP0      ;/
356:                MOVF    adrEE,W         ;EEmenory
357:                MOVWF   EEADR           ;adress to read...
358:                BSF     STATUS,RP0      ;bank 3
359:                BCF     EECON1,EEPGD    ;point to EEprom menory
360:                BSF     EECON1,RD       ;set RD bit to begin read
```

```
361:                BCF     STATUS,RP0          ;bank 2
362:                MOVF    EEDATA,W            ;W = EEDATA
363:                BCF     STATUS,RP1          ;\  Bank 0
364:                BCF     STATUS,RP0          ;/
365:                RETURN
366: ;adresse-pointer værdi skal ligge i W-reg inden kaldet. Z = 1 hvis der er match,  elers er Z = 0 når der returneres.
367: compareEE      MOVWF   adrP
368:                MOVWF   adrEE
369:                CALL    rdEEp
370:                SUBWF   IRtype,W
371:                BTFSS   STATUS,Z
372:                GOTO    exitCEE
373:                INCF    adrP,F
374:                MOVF    adrP,W
375:                MOVWF   adrEE
376:                CALL    rdEEp
377:                SUBWF   IRmode,W
378:                BTFSS   STATUS,Z
379:                GOTO    exitCEE
380:                INCF    adrP,F
381:                MOVF    adrP,W
382:                MOVWF   adrEE
383:                CALL    rdEEp
384:                SUBWF   IRadr,W
385:                BTFSS   STATUS,Z
386:                GOTO    exitCEE
387:                INCF    adrP,F
388:                MOVF    adrP,W
389:                MOVWF   adrEE
390:                CALL    rdEEp
391:                SUBWF   IRdata,W
392: exitCEE        RETURN
393: ;adresse-pointer skal ligge i W-reg inden kaldet.
394: save2EE        MOVWF   adrP
395:                MOVWF   adrEE
396:                MOVF    IRtype,W
397:                MOVWF   dataEE
398:                CALL    wrEEp
399:                INCF    adrP,F
400:                MOVF    adrP,W
401:                MOVWF   adrEE
402:                MOVF    IRmode,W
403:                MOVWF   dataEE
404:                CALL    wrEEp
405:                INCF    adrP,F
406:                MOVF    adrP,W
407:                MOVWF   adrEE
408:                MOVF    IRadr,W
409:                MOVWF   dataEE
410:                CALL    wrEEp
411:                INCF    adrP,F
412:                MOVF    adrP,W
413:                MOVWF   adrEE
414:                MOVF    IRdata,W
415:                MOVWF   dataEE
416:                CALL    wrEEp
417:                RETURN
```