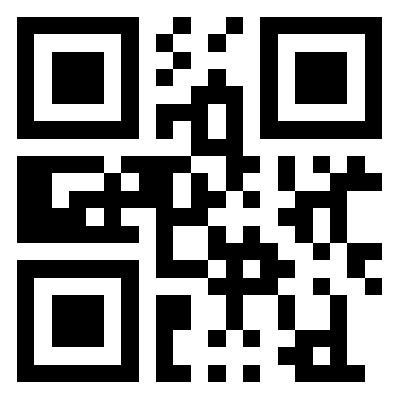
|  |  |  |  |
| --- | --- | --- | --- |
| IST_logo | ***Industrial Processes Automation***  *MSc in Electrical and Computer Engineering*  *Scientific Area of Systems, Decision, and Control*  *Winter Semester 2018/2019* |  | ***Group: C2***  ***70547*** *– João Ferreira* ***75268*** *– Rúben Tadeia* ***75987*** *– João Ribafeita* ***80978*** *– Gonçalo Pedro* |

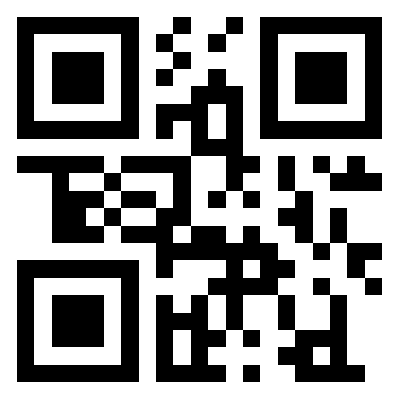
***1st Laboratory Assignment – Part B [[1]](#footnote-2)***

***Alarm System for Intrusion Detection***

**Report questions**

**Q1.** *(Hardware identification)* Identify the PLC inputs and outputs, **from/to the** **keyboard**, to be used on the intrusion detection alarm system.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **PLC Input (chosen variable name)** | **PLC Identifier (physical address)** |  | **PLC Output (chosen variable name)** | **PLC Identifier (physical address)** |
| I\_KB\_LINE\_1 | %i0.2.4 |  | O\_KB\_COL1 | %q4.4.4 |
| I\_KB\_LINE\_2 | %i0.2.5 |  | O\_KB\_COL2 | %q4.4.5 |
| I\_KB\_LINE\_3 | %i0.2.6 |  | O\_KB\_COL3 | %q4.4.6 |
| I\_KB\_LINE\_4 | %i0.2.7 |  |  | |

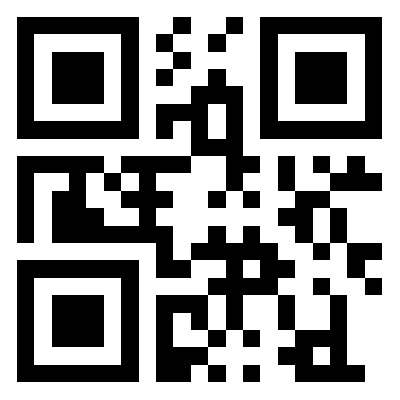
**Q2.** Describe the reading/identifying strategy to be implemented to solve the keyboard reading problem.

**Resposta:** Neste trabalho foi usada uma estratégia de pesquisa por colunas para ler o teclado.

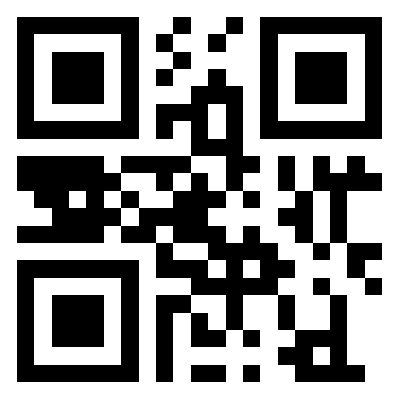
Após chamar a subrotina, com a flag de activação verdadeira, é efectuada uma pesquisa no teclado.

Para realizar a pesquisa, cada uma das colunas do teclado é activada de forma sequencial, por um tempo determinado, de alguns milissegundos. Enquanto a coluna está activada, verifica-se se alguma das linhas está activada; caso uma das linhas esteja activada, é guardada a tecla premida e activada a flag que indica que uma tecla foi premida; caso esta flag esteja activa, a pesquisa pára. Caso não exista uma tecla premida, verificam-se as colunas seguintes, de forma cíclica, até haver uma tecla premida, ou até existir um *timeout*, previamente definido, caso em que a variável M\_key toma o valor ‘-1’.

De modo a evitar erros na pesquisa, causados pela activação sucessiva de colunas, deveríamos ter implementado um temporizador de “espera”, que implementasse um tempo morto entre a desactivação de uma coluna e a activação da coluna seguinte.

**Q3.** Indicate the timer(s) and memory cell(s) used to implement SR\_get\_key.

|  |  |  |
| --- | --- | --- |
| **Memory or timer (chosen variable name)** | **PLC physical address (if specified)** | **Short description** **of the memory or timer** |
| TON\_POLL |  | Timeout do scan ao teclado |
| TP\_Polling |  | Tempo de scan para cada coluna do teclado |
| M\_poll\_timeout |  | Flag que indica timeout do scan (-1) |
| M\_Poll\_COL |  | Cursor que indica a coluna que se deve activar |
| M\_Poll\_Activation |  | Flag para activar o scan ao teclado |
| M\_key\_pressed\_ack |  | Flag que indica que uma tecla foi premida, parando o scan |
| M\_key |  | Variável (INT) que contém a Tecla premida |

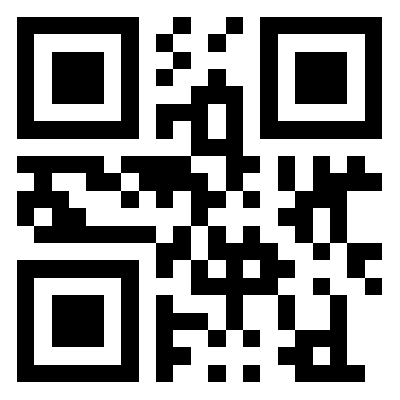
**Q4.** Upload to the PLC and run the program proposed for keyboard reading. Comment on the results obtained.

**Resposta:** Correndo o programa na PLC, verifica-se que, no geral, o mesmo é executado de acordo com as expectativas.

No entanto, existem por vezes alguns erros aquando da leitura de teclas (principalmente em alguns teclados em pior estado), que fazem com que a tecla correcta não seja reconhecida. Estes problemas poderiam ser evitados implementando uma estratégia de *debouncing* mais adequada, que não se baseasse apenas na capacidade existente nos módulos usados no laboratório (por exemplo, usando temporizadores adicionais).

Devido a decisões previamente tomadas na implementação, a subrotina que lê o teclado apenas está activada em momentos específicos, ao invés de estar sempre a ser executada).

Dado que também não implementámos uma flag de *key\_up* (mas sim uma que indica que uma tecla foi premida, em conjunção com um temporizador que coloca a leitura em “pausa” por um pequeno intervalo de tempo, após ler correctamente uma tecla), salvaguardam-se as repetições “acidentais” de teclas, que poderiam preencher o buffer. As flags implementadas permitem ainda que, em caso de conflito (várias teclas premidas ao mesmo tempo), apenas a primeira é detectada; no entanto, uma estratégia mais adequada seria rejeitar as leituras que ocorrem nestas circunstâncias, emitindo um aviso luminoso e/ou sonoro para avisar o utilizador do sucedido.

**Q5.** Use the **logging method indicated in part A question 9** to show changes in the inputs when a person enters the code **1234** using the keyboard. Show **time plots** of the relevant input signals.

**Resposta:** De modo a poder efectuar o *datalogging* das variáveis relevantes, decidimos alterar um pouco o código disponibilizado (quer o código em ST, em que aumentámos a memória disponível para o *datalogging*, quer o código Matlab, em que alterámos um pouco as estruturas dos dados, de modo a acomodar correctamente o incremento da capacidade de *datalogging*). Ainda de modo a permitir registar todas as variações num pequeno intervalo de tempo (devido aos apenas 16 bits disponíveis para registar o número do *scan cycles*, permitindo o registo correcto de cerca de 32 000), os tempos dos temporizadores foram reduzidos.

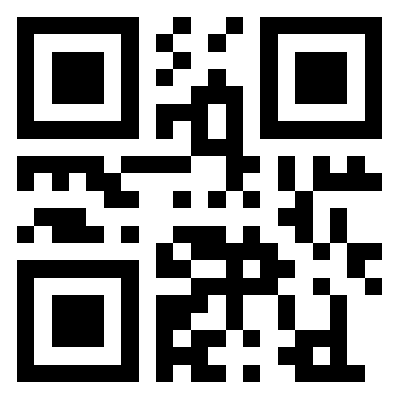
Com estas alterações, foi possível registar mais do que dez mudanças de estado do sistema, permitindo obter uma imagem “global” do funcionamento da rotina.

Seguem os resultados obtidos no *datalogging*:



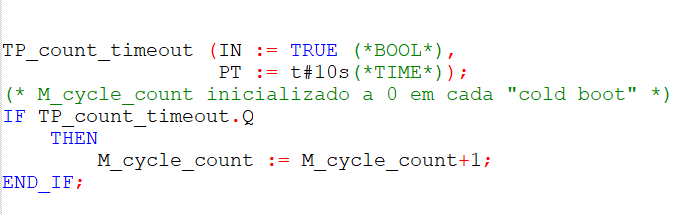
Analisando a figura anterior, verifica-se que o sistema funciona de acordo com o previsto (a maioria das funcionalidades avançadas foi implementada na parte C).

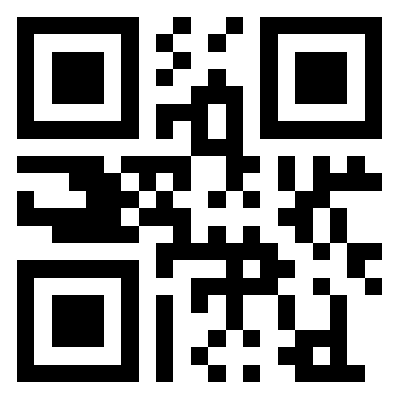
Ao activar o sistema, após o tempo de espera definido, o sistema fica efectivamente activado (momento em que se liga O\_ativo). Caso, enquanto o sistema esteja activado, exista uma detecção de intrusão, a luz de alarme (O\_alarm) é ligada. Após esperar pelo temporizador, é ligado o buzzer (O\_buzzer), ao mesmo tempo que é activada a pesquisa do teclado (M\_poll\_activation); para desligar o alarme, o utilizador introduz o código correcto (neste caso, 1234), seguido de #; o sistema reconhece que um código foi introduzido (M\_code\_inserted) e, como no presente caso o código é válido (M\_valid\_code), o alarme é desligado.

**Q6.** Consult in the manuals how to use **PLC system words** to obtain (i) minimum and (ii) maximum scan-cycle times when the PLC is running just the SR\_get\_key called by one section of MAST. Indicate those times in milliseconds. Another method, write and test one **Structured Text program** based on a timer that counts 10 seconds and counts the number of scan cycles in those 10 seconds.

|  |  |  |
| --- | --- | --- |
| Minimum scan cycle time msec  (use system word - %SW32) | Maximum scan cycle time msec  (use system word - %SW31) | Average scan cycle time msec  (program based on a timer) |
| 1 | 4 | 1.514 |

**Resposta:** Segue o Código usado para implementar a função pretendida:



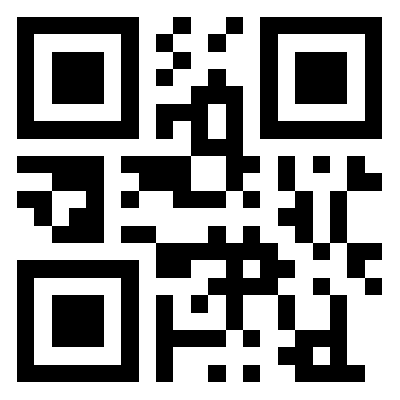
**Q7.** Comment the following statement: "Not waiting for the end of reading a key may imply erroneous string readings".

**Resposta:** Dependendo da forma como está implementada a pesquisa do teclado, podem acontecer diferentes tipos de erros.

No caso de se pesquisar continuamente o teclado (não parando após premir a tecla), pode dar-se o erro de detectar uma linha activada, com a coluna errada activada, o que faz com que possa aparecer qualquer uma das três teclas existentes na linha. Podem ainda acontecer conflitos, quando se carrega em várias teclas.

Na implementação usada, mitigaram-se alguns destes erros, através do uso de uma flag que indica que uma tecla foi premida, parando a pesquisa do teclado nesse momento. Na subrotina SR\_get\_multiple\_keys, foi implementado um temporizador entre scans (bem sucedido) consecutivos ao teclado, de modo a evitar a repetição inadvertida de teclas devido a premir as teclas durante intervalos de tempo longos. Estratégias eficazes de *debouncing* (por exemplo, timers), podem também ser usadas para melhorar a exactidão dos dados obtidos.

Uma estratégia mais robusta consistiria em usar flags que indicassem o fim do evento “tecla premida”, que então permitissem efectuar novas pesquisas ao teclado.

**Q8.** Every hardware setup in the laboratory is based in one of two options, (i) a DMY 28FK input and output module, or (ii) one DEY16D2 input module and one DSY16T2 output module. Consider now a option, where the outputs are implemented by a DSY16R5 module (see in the webpage of the course the item "HW IO Module" pointing to the file "PLC\_3\_HW\_IO\_Discrete.pdf"). What needs to be changed in the keyboard reading strategy?

**Resposta:** Consultando o *datasheet* do módulo DSY16R5 é um módulo de saídas, activadas por relés (ao invés de transístores). Tal tem várias implicações na estratégia a usar para o scan do teclado, maioritariamente devido ao tempo de resposta (mais lento que dos módulos com transístores) e do ciclo de vida dos relés, bastante menor que o dos transístores, caso sejam usados da mesma forma.

Tal implica que se mude a estratégia de scan do teclado: uma estratégia mais adequada seria:

1. Activar as 3 colunas;
2. Verificar se alguma das linhas está activada;
3. Em caso afirmativo, desligar uma coluna de cada vez;
4. Quando a coluna correspondente à tecla premida for desactivada, a entrada deve deixar de estar activa. Caso tal aconteça, foi encontrada a correspondência correcta;
5. Guardar o valor da tecla premida;
6. Voltar a activar as colunas.

A estratégia anterior reduz consideravelmente o número de activações dos relés, quando comparada com a estratégia de “*polling*”. Com a actual estratégia, os relés apenas são activados caso uma tecla seja premida (e na maioria do tempo não há teclas premidas) e, no pior caso, cada um dos 3 relés mudará o seu estado duas vezes.

Considerando um ciclo de vida típico de 1 000 000 ciclos (**IEC 60947**, DC-12: *Control of resistive loads and solid state loads with opto-coupler isolation*), cada um dos relés terá uma vida útil que corresponde, no pior dos casos, a 166 667 eventos de tecla premida, um valor consideravelmente grande.

Caso se usasse a estratégia de *polling*, com os tempos usados anteriormente (25 ms por coluna), cada relé teria uma vida útil de menos de 11 horas (de scans ininterruptos).

No entanto, apesar de para este sistema não ser vantajoso usar um módulo de saídas activadas por relés, o referido módulo pode ser útil em outros sistemas, em que seja necessário activar cargas fortemente indutivas, com transitórios, ou em que seja necessário isolamento galvânico das saídas, entre muitos outros casos).

1. Original guide by Prof. Paulo J. Oliveira. 2019 revision by Prof. José Gaspar. [↑](#footnote-ref-2)