Contents lists available at ScienceDirect

Sensors and Actuators A: Physical

journal homepage: www.elsevier.com/locate/sna

Letter to the Editor

Code reading synchronization method for pseudorandom position encoders

ARTICLE INFO

Keywords:
Position encoders
Pseudorandom binary sequences
Encoding

ABSTRACT

A new method for code reading synchronization for application to pseudorandom position encoders is proposed. This scanning method uses code track itself for synchronization. The Manchester encoding of pseudorandom code bits is applied.

1. Introduction

Pseudorandom position encoders enable absolute position measurement using one-bit width code track [1] due to which they are very actual today. They can also be found on the market under the name of virtual absolute encoders. The well-known property of pseudorandom sequences is used [1,2] in the sense that two n-tuples for two consecutive positions contain an identical array of $(n-1)$ bits, so the code can be written along one code track, Fig. 1. From the aspect of pseudorandom position encoder's realization, the key problems are code reading [3], code reading synchronization or code scanning in the sense of reliable code reading moment defining [4,5], code reading errors detection [6] and pseudorandom/natural code conversion [1,5]. The possibility that n-bits of the position code are not read in parallel but assembled sequentially in an n-bit bidirectional shift register using one [1] or two sensor heads [3], is one more advantage of pseudorandom encoders. But, there are solutions which are based on parallel reading by applying of n sensor heads [5] and parallel reading by applying linear photodetector array [7,8]. Pseudorandom coding is also very suitable for two-dimensional position measurement [9].

In this paper, there will be considered problems related to code reading synchronization which is important with all position encoders including pseudorandom encoders [4]. Elimination of the code reading ambiguities is necessary in order to get correct position information. There are two approaches for this problem. Code track scanning can be directly realized by using external synchronization implemented as an additional physical track which lies along the pseudorandom code track and guide-path [1,4]. The other way introduces internal synchronization and requests applying additional incremental encoded free-running wheel [5]. Introducing of additional track is avoided, but due to the used incremental technique the error acumulation is possible, and so often resynchronization of used logic is needed, and it is the main drawback of this method. For a long period it was thought that basic scanning methods are defined well enough and that there is a necessity of adding a new element in the form of a physical track or incrementally coded wheel. A new method which does not request anything additional is proposed here, and uses only the code track itself for synchronization of the code reading.

2. New code reading syncronization method

Pseudorandom binary code is unsuitable for scanning method realization due to unpredictable series of logic 0 and 1. Therefore, Manchester coding of pseudorandom code bits is suggested, as described in [7]. So, logic 1 is replaced with sequence "10" and logic 0 with sequence "01". In case of optical implementation, Fig. 1, the outlook of code track is shown with the arrangement of applied sensor heads. Manchester coding is applied in [7] for a different reason with the goal of combining methods of absolute and incremental position measurement. In this paper, the goal is to get a larger number of transitions between logic 0 and 1 and so with the help of appropriate synchronization logic to provide synchronization pulses. Those pulses feed clock input of reading buffer [5] for the case of parallel code reading or clock input of bidirectional shift register for the case of serial code reading [3]. In that way the synchronization pulses, termed as "LOAD" pulses, define a moment of reliable code reading, and that is when heads are located on the middle of the segments which are being read.

Synchronization head AUT is always necessary when synchronization is performed by physical track. Abbrevation AUT is related to the word ambiguities, which is widely accepted due to the role of synchronization head during elimination of the code reading ambiguities. Synchronization head AUT is located exactly in the middle of quantization step d, for the given position of sensor heads $x(n)$ and $x(1)$, Fig. 1. Logic value on the output of the sensor head $x(n)$ corresponds to the read bit of pseudorandom code S(A), while on the output of the sensor head $x(1)$ an invertor gate is necessary to be applied. Wherever it is possible to provide here defined arrangement of sensor heads, suggested synchronization method can be applied, and only small technical adjustments are necessary. In the case of serial code reading with two sensor heads [3] there is a need to reduce the distance between the existing sensor heads from $n{\cdot}d$ to $(n-1){\cdot}d+d/2$, Fig. 1. Similarly, at parallel code reading with n sensor heads [5] it is necessary to move one head for $d/2$, so that exist two reading heads on the defined distance. At serial reading with one sensor head $x(n)$ it is necessary to introduce a sensor head $x(1)$, Fig. 1. In any case, for many applications it is more acceptable to introduce one additional sensor head instead of additional physical track. Additionally, the conditions for applying of code reading

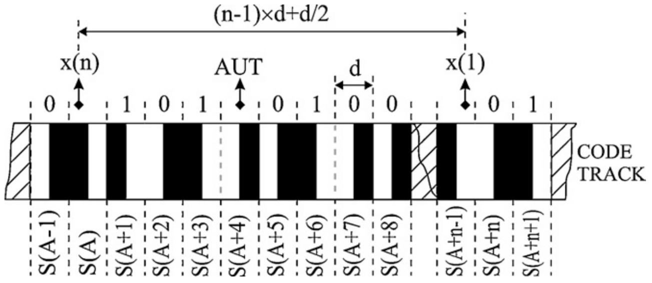

Fig. 1. Sensor heads arrangement in new synchronization method of the pseudo-random code reading.

correctness checking method described in [6] are provided by that.

Further in the text the functioning of suggested method is described, Fig. 1. Let the movable system (MS) move to the right. After passing $d/4$ of the path, the sensor head $x(n)$ will be located in the middle of the absolute scale sector, whose width is d. In that moment a transition from one to the other logic level will be detected and using an edge detector an electrical pulse can be obtained. At the same time, the head $x(1)$ is located at the transition between two sectors which does not always lead to, as is the case here, the changing of the logic level at its output. After the next $d/4$ of the path passed, synchronization head AUT will be at the transition between the two sectors and on the output of the edge detector, for the given bit sequence, a pulse will be obtained. This pulse may not be accepted as synchronization. It will not appear on the transition of every two sectors of an absolute measurement scale, which additionaly complicates realization of a mechanism for its elimination.

Furthermore, after $d/4$ of the passed path, the sensor head $x(1)$ will be located in the middle of the next sector and there will be the logic level changing of the electrical signal on its output. Sensor head $x(n)$ will be located on the transition between two sectors and in this case (it is not always the case) on the output of the signal edge detector there will appear a pulse. After the next $d/4$ of the path passed, which completes movement of one absolute quantization step d, the head AUT will be located in the middle of the next sector. A pulse is always detected at that transition and that pulse should be accepted as synchronization “LOAD” pulse. This means that during the movement of MS to the right, between two “LOAD” pulses, the following sequence appears:

- obligatory pulse on the output of sensor head $x(n)$ (more precisely on the output of the signal edge detector) and eventual pulse on the output of head $x(1)$,
- possible false “LOAD” pulse on the output of the sensor head AUT,
- obligatory pulse on the output of the sensor head $x(1)$ and possible pulse on the output of the head $x(n)$.

In the case of permanent movement of the MS to the left, the sequence would be opposite in relation to the one noted above. It can be concluded that independently on the movement direction, two pulses on the output of the OR gate are always obtained, between two synchronization “LOAD” pulses, while outputs of the signal edge detectors of the code reading heads feed its input. This can be used for the elimination of the false synchronization pulse.

Otherwise, due to the apllied Manchester coding of pseudo-random code bits, information about movement direction can be achieved in a classical way by adding one more sensor head [3], fixed on the distance of $d/4$ from the sensor head AUT. Other methods are also possible, such as detection of the movement direction changing based on the already noted pulse sequence from sensor heads. In that case using a more complicate logic circuit is requested and one such solution is applied during testing of here proposed method. We here admit to the fact that it can be relatively simple to come to information about movement direction of MS ($RGT = 1$ – movement to the right).

Table 1
Definition of logic functions D_1 and D_2.

RGT	Q_1	Q_2	D_1	D_2
1	0	0	0	1
0	0	0	0	1
1	0	1	1	0
0	0	1	1	0
1	1	0	0	1
0	1	0	0	1
1	1	1	1	0
0	1	1	1	0

3. Synchronization logic

For elimination of the false synchronization pulse, one bidirectional two-bit counter is needed. For the case of D flip flop usage, table of states on flip flop inputs just before arriving of clock pulse is given in Table 1.

The following logic functions are obtained by Karnaugh maps using:

$$D_1 = RGT \cdot (\overline{Q_1} \cdot Q_2 + Q_1 \cdot \overline{Q_2}) + \overline{RGT} \cdot (\overline{Q_1} \cdot \overline{Q_2} + Q_1 \cdot Q_2)$$

$$D_2 = \overline{Q_2}$$

By further minimization the following is obtained:

$$D_1 = RGT \oplus \overline{Q_1} \oplus Q_2$$

Complete realization of the control logic for code reading synchronization is shown on Fig. 2. Additional D flip flop serves for synchronization establishment after system initialisation and after every detected error. Start pulse denotes the begining of the synchronization establishing, resets D flip flop and sets output of bidirectional two-bit counter in state $(\overline{Q_1}Q_2) = (1, 0)$. Every pulse obtained by sensor head AUT resets the counter and by means of D flip flop examines output L (L = “1” at $(\overline{Q_1}Q_2) = (0, 0)$). In order for L = “1”, it is necessary that after the pulse from head AUT (for the same movement direction of MS), there appear two successive pulses from heads $(x(n), x(1))$. Those pulses present clock pulses for the counter. It will happen when the false synchronization pulse does not appear. It does not appear when bits which correspond to the previous and current absolute sectors are different. That is after a few passed quantization steps d in the same direction. After L = “1”, the first pulse from the sensor head AUT is accepted as synchronization. D flip flop is “frozen” into the state Q = “1”, and in this way it is signaled that code reading synchronization is established. Further, counter counts pulses from the output of logic for pulse obtaining by means of heads $(x(n), x(1))$ and always when it comes into the state $(\overline{Q_1}Q_2) = (0, 0)$, allow pulse accepting from the sensor head AUT for synchronization. If that pulse appears before counter exiting from this state, a counter reseting is made, and its state now is $(\overline{Q_1}Q_2) = (1, 0)$. It means, when Q = “1” counter reseting is made only with the “LOAD” pulse.

The proposed synchronization logic ensures reliable synchronization pulse and at multiple changing of MS moving direction between two “LOAD” pulses. By this, planned hysterezis characteristics of the absolute position measurement is also provided. After one accepted “LOAD” pulse, the state of the counter becomes $(\overline{Q_1}Q_2) = (1, 0)$ and in that way the acceptance of possible repeated “LOAD” pulses in the same sector is prevented. This synchronization logic can be applied in all pseudorandom encoders. In the case of