Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2873476 A1 2016/06/08

(21) 2 873 476

(12) DEMANDE DE BREVET CANADIEN
CANADIAN PATENT APPLICATION

(13) A1

(22) Date de dépôt/Filing Date: 2014/12/08

(41) Mise à la disp. pub./Open to Public Insp.: 2016/06/08

(51) Cl.Int./Int.Cl. *G09G 5/39* (2006.01),
H04N 5/44 (2011.01)

(71) Demandeur/Applicant:
IGNIS INNOVATION INC., CA

(72) Inventeurs/Inventors:
UNKNOWN, ZZ;
GHOLAMREZA, CHAJI, CA;
YASER, AZIZI, CA

(74) Agent: MARKS & CLERK

(54) Titre : ARCHITECTURE D'AFFICHAGE DE PIXELS INTELLIGENTS
(54) Title: SMART-PIXEL DISPLAY ARCHITECTURE

(57) Abrégé/Abstract:
This invention covers techniques for emissive display systems constructed on integrated architecture platform. In this architecture, the pixels are smart and can behave differently under different conditions to save power, provide better image quality, and conserve their value to reduce the power consumption associated with programming.

1. Field of Invention

This invention covers techniques for emissive display systems constructed on integrated architecture platform. In this architecture, the pixels are smart and can behave differently under different conditions to save power, provide better image quality, and conserve their value to reduce the power consumption associated with programming.

2. Smart-pixel Display Architecture

A display system with monolithic architecture is illustrated in Figure 1. This architecture is constructed of a front-end interface, Gate and Clock-Drivers, and in-pixel driving elements.

The front-end interface can include a timing controller (TCON) and readout circuitry (ROC) and/or a data driver. The front-end further networks with an array of in-pixel driver elements and gate/clock-drivers. The gate/clock-driver provides control and clock signals to rows of pixel elements. Each in-pixel driver element is composed of a controller, memory, current/voltage driver, and a light-emitting device (EL).

The controller within each pixel element supervises the flow of data in the memory devices based on the command signals on the WR and CLK lines.

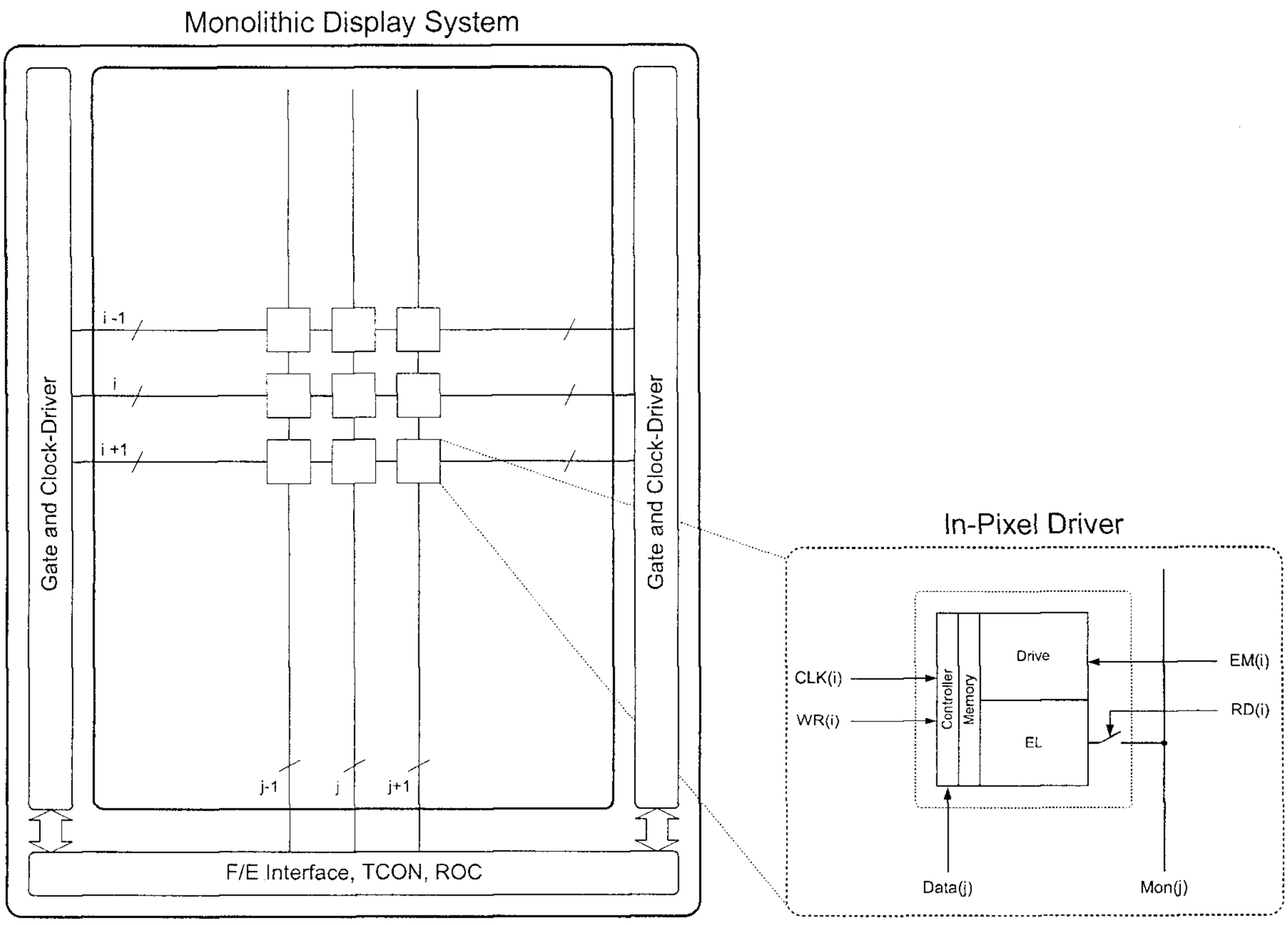

Figure 1: Monolithic display system architecture.

All the loading operation explained here can be applied to other structure in the document, and also other possible structures not explained in this document. In addition, one can easily take some feature of one method and mix it with other methods. The examples here are for demonstration and not inclusive of all possible cases.

In one aspect of the invention, the data is stored in registers, connected to the columns lines, from the video interface. Then the data is loaded from these registers in parallel or serially. In Figure 2, the column line can be multi-bit to transfer more data during each clock.

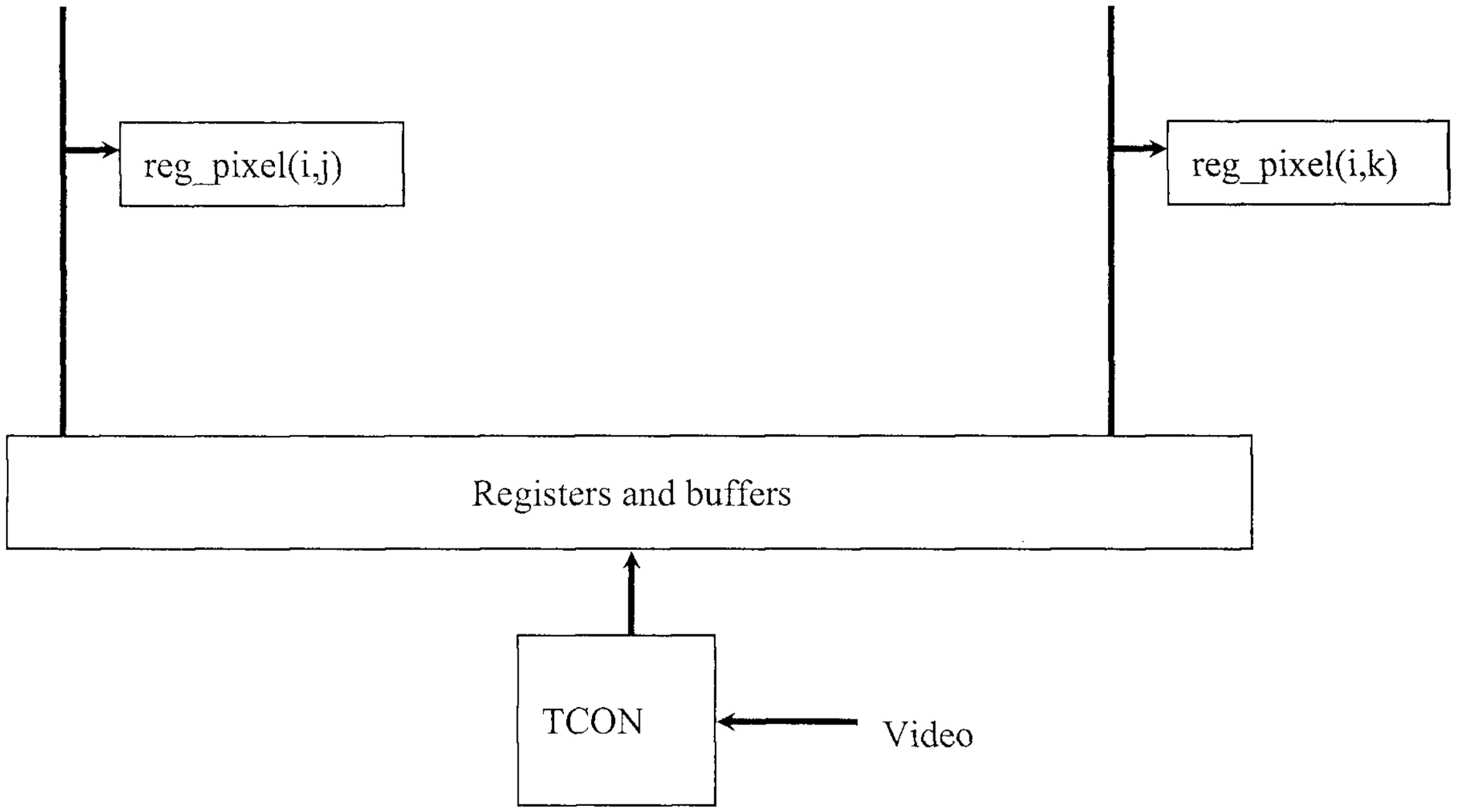

Figure 2: An example of datapath between video interface and pixel memory.

In another aspect of the invention, the data is stored in said registers partially and then the partially loaded data is transferred to the pixel in parallel or serially. In this case, the registers at the boundary of the display will have fewer bits compare to the row data. In one example, it can only have one bit for each pixel (if the row has 240x3 pixels, the total bit for the boundary registers will be 720 instead of 720xdata_width where data width is the number of bit for gray scales, e.g. 8 bits). Here, the first bit of each pixel is loaded into the boundary registers, after that the data is transferred to the pixel memory (pixel_reg). This operation continues till all the data is loaded into the pixels of said row. Then the operation is repeated for the next row. One can load the first bit of more than one row first and then move to the next bit. In this case, one can turn on the display (or row) after each bit and then load the next bit and turn on the display and so on. The ON time of the pixel will be defined based on the weight of each bit loaded into the row (or display).

In another aspect of the invention, the data is directly loaded into the pixel memory from the video interface (Figure 3). Here, the pixel memories in a row form one or more shift register chains during the programming time, and the data from the interface is loaded into the shift registers.

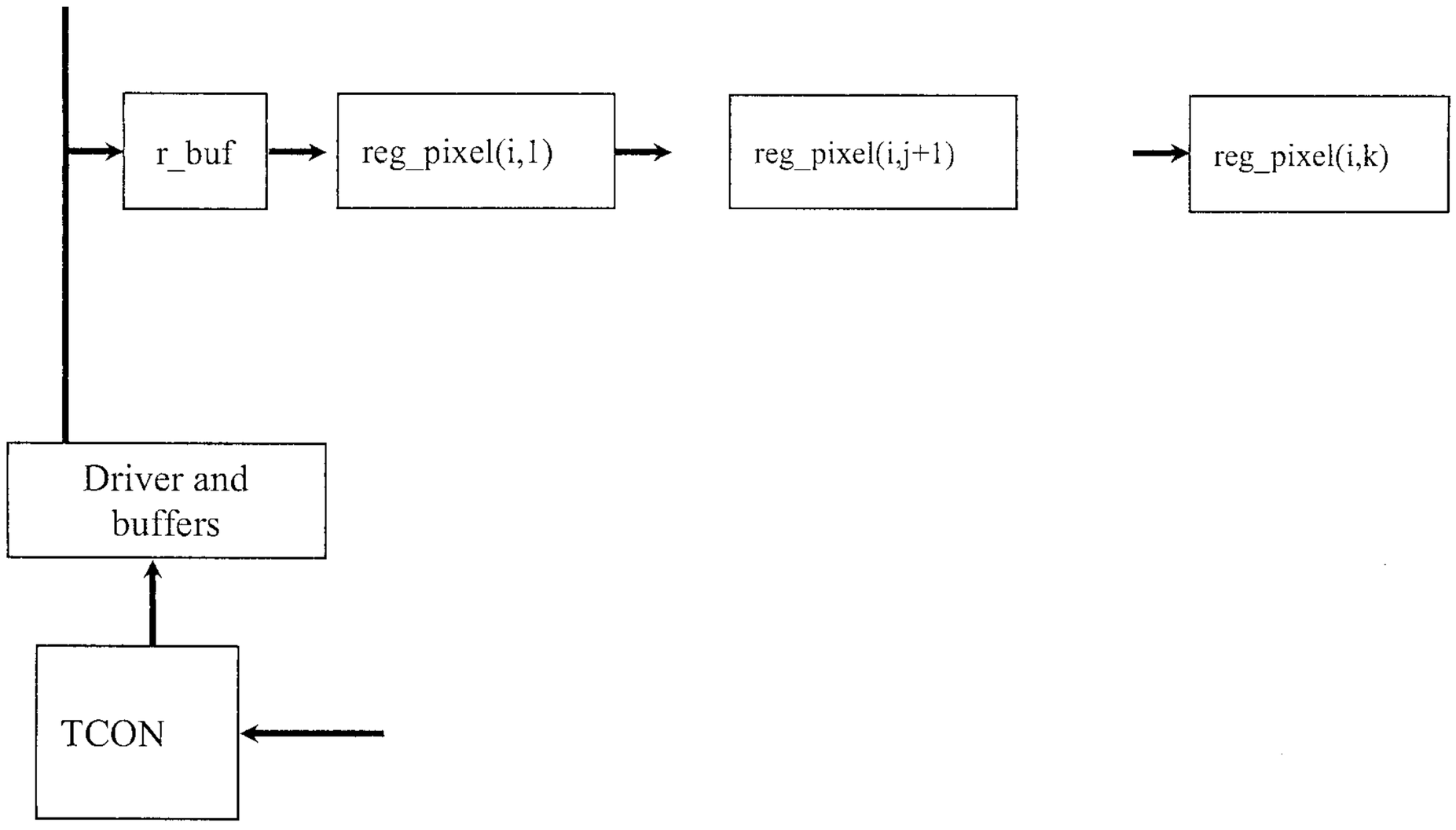

Figure 3: Another example of data path between video interface and pixel memory.

Here, the r_buf can include a switch that disconnect the data line form the rows that are not selected for programming. Also, it can have some conversion functionality such as converting low voltage differential signals to normal swing signals. Also, the driver and buffer can do part or all of the conversion and so the r_buf block does the remaining part.

To avoid reprogramming the pixels during each frame independent of if their data changed or not, a controller is added to each pixel. Here an independent signal through this controller can enable or disable pixel programming. However to reduce the number of the signals, the data can begin with some value that tells the controller to enable or disable the programming. For example, the first bit can identify the programming mode of the pixel. After that if the pixel is in reprogramming mode, the data will be saved in the shift register. If the pixel is in halt mode (saving its previous data), the data in shift register are not updated. As a result, the data for that pixel can stay as it is and so no refreshing power consumption will be associated with that pixel circuit.

In case, the data is loaded through the row shit register, the data can be first loaded to the controller to define the operation of each pixel and then the data is loaded to the shift register chained formed by pixel memories. If a pixel does not need to be reprogrammed, the controller can bypass it in the shift register chain.

The drive element in the pixel can be a fixed current/voltage or it can be changed depending to the display operation conditions and/or depending on the weight of the bit applied to the pixel.

One example of the display operation can be peak brightness. In this case, if the pixel brightness increases, the driving force of the pixel can increase to accommodate the peak brightness without loosing digital grey levels. In another case, the driving force of the pixel is adjusted based on the weight of the bit applied to it. In another case, the pixel driving force is adjusted based on group of the bits.

In one example, the pixel operation condition changes to adjust the drive force. For example, the bias condition of the driver can be adjusted to either apply higher voltage or higher current to the emissive device when is needed. In another case, multiple drivers with different strength exist in the pixel. Each of these driver elements is controlled by different bits of grayscales or they are controlled by global signals based on display performance requirements.

3. In-Pixel Driving Element (pixel driver)

The in-pixel driving element (pixel driver) can be either a voltage based driver or a current based driver. In case of voltage driver, a simple switch can connect the voltage to the emissive device (light-emitting device). This can be one switch connected to a controllable/fixed voltage bias or multiple switches connected to multiple bias voltages.

In another example, the pixel driver is a current driver. Here, the gray scale bit either controls the strength of the current output of the pixel driver; or control the connection of the pixel driver to the emissive device; or it enables/disables the current driver. In another example, one can easily mixes the three operational modes to take advantage of best characteristics of all of them.

An example implementation of in-pixel driving is illustrated in Figure 4. A programmable current source (I_{Pix}) provides the driving current for the light-emitting device (EL).

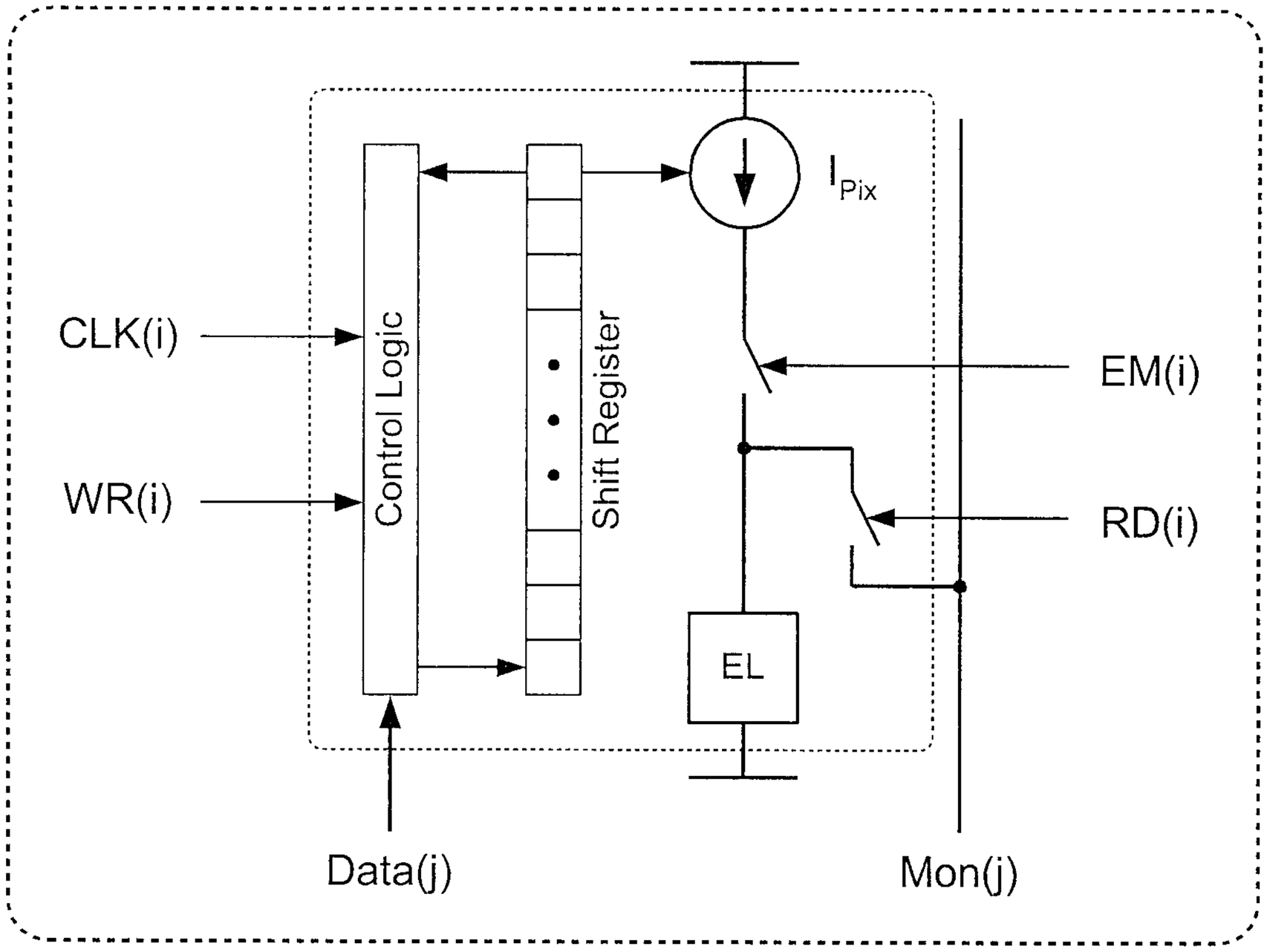

Figure 4: In-pixel driving element

An EM switch can be used to disconnect the pixel driver from the emissive device. Also, an RD signal provides a signal path to steer the pixel current/charge towards the ROC. This signal can be shared with other signals in the pixel. Or the controller can control this signal based on the operation mode of the pixel and status of other signals.

In case the signal is defined by the strength of the output current, the grayscale store in the shift register selects different strength for the output current. In this case, the current source has different elements with different output current strength. And different combination of this current levels are applied to the emissive device according to the data stored in the shift register. Similar method can be applied to the voltage-based driver.

In another case, the current source has a fixed output. In this case, the gray scales are defined based on the time the pixel is ON which is controlled by the data stored in the shift register. In one case, the data stored in shift register is compared with a counter value. When the two values are the same the pixel current is off (or the current source is disconnected from the emissive device; or its current is redirected to another root). It is worth mentioning that one can do the reverse of aforementioned operations without affecting the pixel performance. In one example, when the data in shift register of the pixel is the same as the counter value, the pixel turns ON instead of turning OFF. Here the counter can be non-linear to accommodate the non-linear gamma curves. For example, it counts faster at lower grayscales and slows down as its value increases. The speed of the counter can be function of the gamma curve. In another case, the

output of shift register is connected to the pixel driver (this signal can either enable/disable the current source, or connect/disconnect the current source from the emissive device). Every clock shifts the value of the shift-register. As a result, depending on the value of every bit in the shift register, the pixel driver status can be different. The period of the clocks can be different based on the weight of its corresponding bits in the gamma curve. One can use rotating shift register. In this case, the bit that is shifted out is shift back to the pixel from the other side. As a result, the value programmed in the shift register is preserved and so one can stop refreshing the panel without loosing the content. This can save power consumption associated with display programming for each frame.

In addition, one can use dynamic weight for each bit so that the error associated with time modulation effect is reduced. For example, in one case, bit0 can have the lowest value and so the last clock will have the period time associated with that during the frame time. In another case, bit3 can have the lowest value and so the third clock from the last will have the time associated with the lowest bit during the frame time. This will reduce the contouring artifact caused by time modulation.

In another aspect of this invention, one can use combination of different signal strengths and timing conditions. An easy example of this case is to have few output strength for each pixel. Depending on the condition of the pixel, one of these outputs is used for time modulation. For example, a global signal can identify high brightness mode, and so the highest output strength is used for time modulation driving.

In case of using shift register in the pixel for creating the time modulation effect, the time-modulation clock can be passed to each row through a shift register at the edge of the panel that has similar size as the number of rows or more. The clock pattern that has the weight of each bit is shifted into the shift register after each shift register clock (this clock can be similar to the clock used for creating the select line for each row, which has a period equal, or smaller than the row time. In another example, the clock can be a separate clock. In this case, one can create different time modulation without being limited to the clock period).

Figure 5 demonstrates one example of this operation. Here, the time-modulation clock is generated with timing controller or passed by external circuit to the display. The first part of the clock is not active which is associated with the pixel programming time. After the row programming is finished, the row can be activated (here the clock is active high but it can be active low as well). Then the clock toggles so that it shifts the value in the pixel shift registers one bit forward. Then it stays active for another period of time. The same situation follows for the next row and the row after.

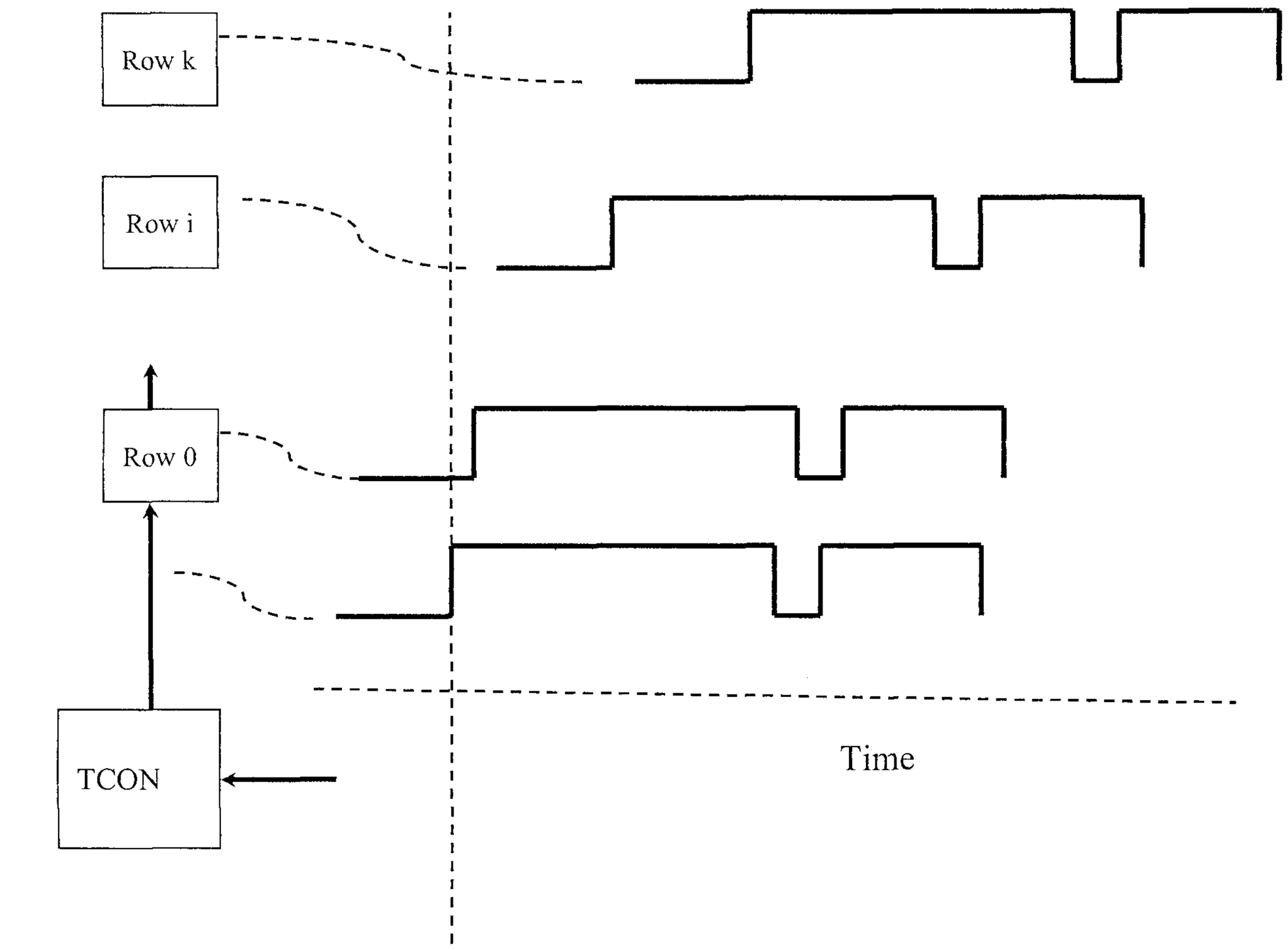

Figure 5: An example of distributing time-modulation clock between rows.

4. In-Pixel Driving Scheme

An example of driving scheme is sketched in **Error! Reference source not found.**. In this scheme, the drive current representing the desired output luminance grayscale is quantized by an N-bit digital signal. The N-bit data is programmed and stored in the shift register of Figure 4. Each bit of the N-bit data ($b_{N-1}b_{N-2}...b_1b_0$) modulates the fixed drive current (I_{Pix}) in a window of time, which is proportional to $2^i \times T_u$ where i is the bit order (0 to N-1) and T_u is the unit time window. Accordingly, the effective EL drive current in each frame time is given by:

$$I_{eff} = I_{Pix} \frac{T_u}{T_{Frame}} \sum_{i=0}^{N-1} b_i 2^i \tag{1}$$

Note that:

$$T_u = \frac{T_{Frame} - T_{prog}}{2^N} \tag{2}$$

and hence replacing (2) in (1) results in:

$$I_{eff} = \alpha \frac{I_{Pix}}{2^N} \sum_{i=0}^{N-1} b_i 2^i \tag{3}$$

where α is a constant given by:

$$\alpha = \frac{T_{Frame} - T_{prog}}{T_{Frame}} \tag{4}$$

During the program time, the driving current is momentarily deactivated by the EM signal. A logic "1" is asserted on the data line and stored in the controller by a clock pulse on the WR in preparation of a program sequence. An N-bit serial data is then clocked in and programmed in the shift register. Finally, a logic "0" is asserted on the data line and stored in the controller by a clock pulse on the WR in order to halt the program mode.

The described sequence along with the proposed in-pixel driving element provides a unique feature, which enables programming of individual pixels in the selected row. This is particularly useful for power saving when only parts of an image are required to be updated in a given frame.

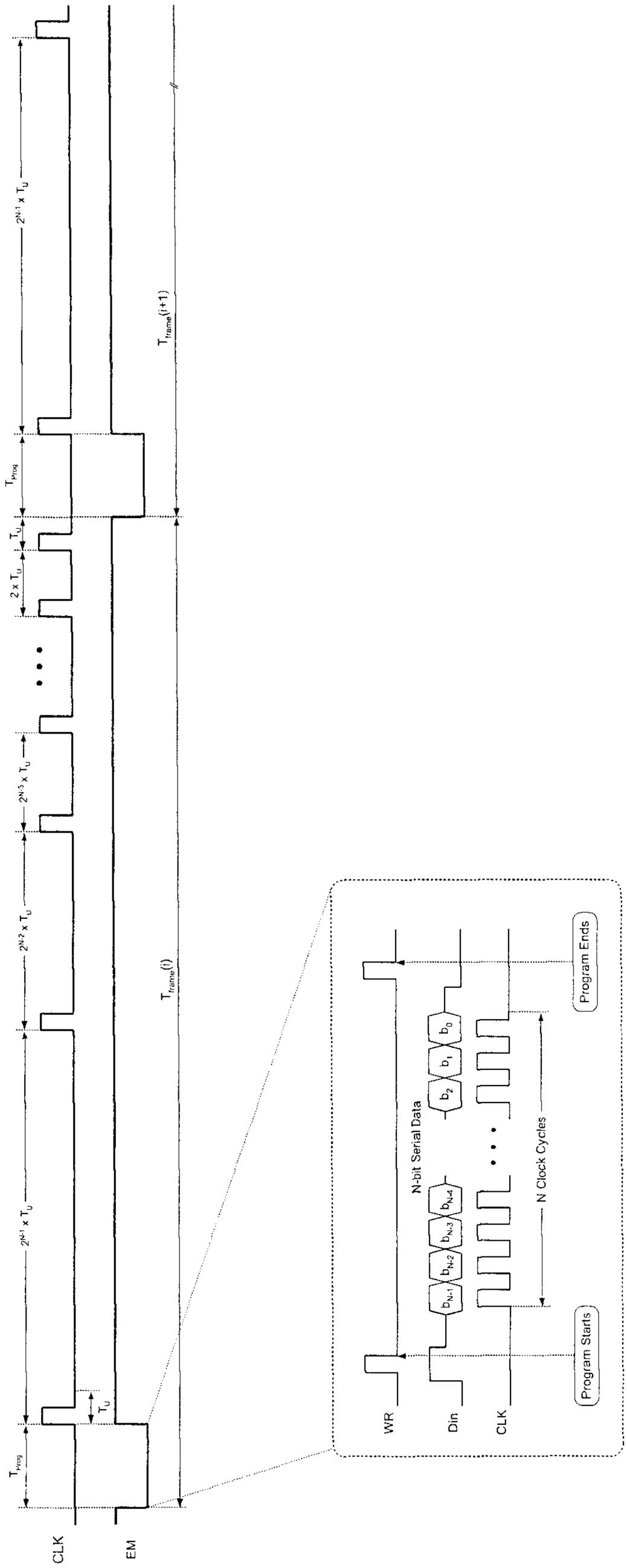

Figure 6: An example driving scheme for in-pixel driver.

5. Low power mode

In this case, the number of gray scales is reduced.

For programming, either the data is being copied in other unused bits of pixel shift registers or part of the shift registers is removed from the chain and so only the required bits are active.

At the same time the number of clock cycle associated with time-modulation clock can be reduced. However, it is not required for functionality of the display to reduce it. It will only save power consumption.

If counter is used for creating time modulation, the counter size is reduced as well to match the new number of gray scales.